UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Renato Moraes Silva

Da Navalha de Occam a um método de categorização de textos simples, eficiente e robusto

Campinas
2017

Renato Moraes Silva

Da Navalha de Occam a um método de categorização de textos simples, eficiente e robusto

Tese de Doutorado apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Doutor em Engenharia Elétrica, na Área de Automação.

Orientador: Prof. Dr. Akebo Yamakami
Coorientador: Prof. Dr. Tiago Agostinho de Almeida

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA TESE DE DOUTORADO DEFENDIDA PELO ALUNO RENATO MORAES SILVA E ORIENTADA PELO PROF. DR. AKEBO YAMAKAMI.

Campinas
2017

Agência(s) de fomento e nº(s) de processo(s): CNPq, 141089/2013-0
ORCID: http://orcid.org/0000-0001-6687-8981

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Luciana Pietrosanto Milla - CRB 8/8129

Si38n

Silva, Renato Moraes, 1988-
Da Navalha de Occam a um método de categorização de textos simples, eficiente e robusto / Renato Moraes Silva. – Campinas, SP : [s.n.], 2017.

Orientador: Akebo Yamakami.
Coorientador: Tiago Agostinho de Almeida.
Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Aprendizado de máquina. 2. Reconhecimento de padrões. 3. Comprimento mínimo de descrição (Teoria da Informação). I. Yamakami, Akebo,1947-. II. Almeida, Tiago Agostinho de. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: From the Occam's Razor to a simple, efficient and robust text categorization approach
Palavras-chave em inglês:
Machine learning
Pattern recognition
Minimum description length (Information Theory)
Área de concentração: Automação
Titulação: Doutor em Engenharia Elétrica
Banca examinadora:
Akebo Yamakami [Orientador]
Sahudy Montenegro González
João Paulo Papa
Levy Boccato
Romis Ribeiro de Faissol Attux
Data de defesa: 14-03-2017
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA — TESE DE DOUTORADO

Candidato: Renato Moraes Silva **RA:** 114927
Data da Defesa: 14 de março de 2017

Título da Dissertação/Tese: "Da Navalha de Occam a um método de categorização de textos simples, eficiente e robusto"

Prof. Dr. Akebo Yamakami (presidente, FEEC/UNICAMP)
Prof. Dr. Sahudy Montenegro González (DC/UFSCAR – Campus de Sorocaba)
Prof. Dr. João Paulo Papa (FC/UNESP/Bauru)
Prof. Dr. Levy Boccato (FEEC/UNICAMP)
Prof. Dr. Romis Ribeiro de Faissol Attux (FEEC/UNICAMP)

A ata de defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no processo de vida acadêmica do aluno.

Dedico esta tese ao meus amados pais, David e Maria, e aos meus queridos irmãos, Leandro e Bruno.

Agradecimentos

Agradeço,

à Deus por abençoar minha vida;

ao meu orientador e co-orientador, Profs. Akebo Yamakami e Tiago A. Almeida, por compartilharem seu conhecimento, pela paciência, críticas construtivas, orientações e motivações que foram fundamentais para a realização deste trabalho e ajudaram a ampliar imensamente minha bagagem profissional;

aos meus pais David e Maria pela educação, por terem me ensinado o valor do trabalho e a correr atrás dos meus sonhos, por acreditarem mais em mim do que eu mesmo, por terem sempre me apoiado e incentivado e pelo amor e carinho que me deram em todas as etapas da minha vida;

aos meus irmãos Leandro e Bruno pelo apoio, pelo companheirismo e por serem sempre meus melhores amigos;

à minha namorada Simone pelo carinho, apoio, paciência e compreensão;

à prof.ª Tatiana A. Pazeto por ter me incentivado a entrar na pós-graduação e pelos seus ensinamentos e troca de experiências que foram fundamentais para minha trajetória no mestrado e doutorado;

ao Prof. Ricardo C. L. F. Oliveira pelo exemplo de humildade e pelas conversas valiosas durante os cafés diários que ocorreram por toda minha pós-graduação;

aos meus colegas do Departamento de Sistemas e Energia pela ótima convivência e troca de experiências;

aos professores Jacques Wainer, Fernando José Von Zuben, Romis Ribeiro de Faissol Attux, Ricardo R. Gudwin, Akebo Yamakami e Takaaki Ohishi pelas ótimas disciplinas oferecidas durante meu mestrado e doutorado;

à agência de fomento CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) pelo apoio financeiro concedido durante todo o período de doutorado;

à Faculdade de Engenharia Elétrica e de Computação e a Universidade Estadual de Campinas pela ótima estrutura que oferece aos estudantes e pesquisadores.

"Nós só podemos ver um pouco do futuro, mas o suficiente para nos darmos conta de que há muito a ser feito"
(Alan Turing)

Resumo

Categorização de textos é um problema que tem recebido muita atenção nos últimos anos devido ao aumento expressivo no volume de informações textuais. O processo manual de categorizar documentos de texto é cansativo, tedioso, demorado e muitas vezes impraticável quando o volume de dados é muito grande. Portanto, existe uma grande demanda para que esse processo seja realizado de maneira automática através de métodos computacionais. Embora vários métodos já tenham sido propostos, muitos sofrem com o problema da maldição da dimensionalidade ou apresentam alto custo computacional, inviabilizando seu uso em cenários reais. Diante disso, esta tese apresenta um método de categorização de texto baseado no princípio da descrição mais simples, nomeado `MDLText`, que é eficiente, rápido, escalável e multiclasse. Ele possui aprendizado rápido, incremental e é suficientemente robusto para evitar o problema de sobreajustamento, o que é altamente desejável em problemas reais, dinâmicos, *online* e de grande porte. Experimentos realizados com bases de dados reais, grandes e públicas, seguidos por análises estatísticas dos resultados, indicam que o `MDLText` oferece um excelente balanceamento entre poder preditivo e custo computacional. Diante desses bons resultados, foi proposta uma generalização inicial do método para lidar também com problemas não-textuais, o que resultou em um método de classificação, nomeado `MDLClass`, que é simples, rápido e pode ser aplicado em problemas binários e multiclasse. A análise estatística dos resultados indicou que ele é equivalente à maioria dos métodos considerados o estado-da-arte em classificação.

Palavras-chaves: aprendizado de máquina; reconhecimento de padrões; categorização de texto; classificação; princípio da descrição mais simples.

Abstract

Text categorization has received attention in recent years because of the ever-increasing volume of text information. For a large number of documents, a manual classification is tiresome, tedious, time-consuming, and impractical, making computational methods attractive to deal with this task. The available methods that address this problem suffer from their computational burden and the curse of dimensionality, undermining their applicability in real scenarios. To overcome this limitation, we propose a simpler, faster, scalable and more efficient classification method based on the minimum description length principle, named `MDLText`. Its incremental and faster learning process makes it suitable to cope with data overfitting, which is desirable for real and large-scale problems. Experiments performed on real, public, and large-scale datasets followed by statistical analyses indicate that the `MDLText` provides an excellent trade-off between predictive capability and computational cost. Motivated by these results, we propose a generalized method, named `MDLClass`, to encompass non-textual problems. Similar to `MDLText`, this extension is simple and fast, and can also be applied to binary and multiclass classification problems. Statistical analyses show that `MDLClass` is equivalent to most of the state-of-the-art classification methods.

Keywords: machine learning; pattern recognition; text categorization; classification; minimum description length principle.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ABC	atributos baseados no conteúdo
ABL	atributos baseados nos *links*
ABLT	atributos baseados nos *links* transformados
χ^2	*chi-square* (chi-quadrado)
BIC	*Bayesian information criterion* (critério de informação Bayesiana)
B.NB	*naïve* Bayes Bernoulli (Bayes ingênuo Bernoulli)
CART	*classification and regression trees* (árvores de classificação e regressão)
CF	*confidence factors* (fatores de confidência)
CSV	*comma separated values* (valores separados por vírgulas)
DFS	*distinguishing feature selector* (seletor de atributos distintivos)
DT	*decision trees* (árvores de decisão)
GSS	Galavotti-Sebastiani-Simi
IG	*information gain* (ganho de informação)
IM	*instant messages* (mensagens instantâneas)
IQR	*interquartile range* (amplitude interquartil)
KNN	*k-nearest neighbours* (k-vizinhos mais próximo)
MAP	*maximum a posteriori* (máximo a *posteriori*)
MCC	Matthews *correlation coefficient* (coeficiente de correlação de Matthews)
M.NB	*naïve* Bayes multinomial (Bayes ingênuo multinomial)
MDL	*minimum description length* (descrição mais simples)
MML	*minimum message length* (mínima mensagem)
NGL	Ng-Goh-Low
NB	*naïve* Bayes (Bayes ingênuo)
OGD	*online gradient descent* (gradiente descendente *online*)
OHE	*one-hot encoding* (codificação de um *bit* por estado)
OR	*odds ratio* (razão de chances)
PDF	*portable document format* (formato de documento portátil)
RF	*random forest* (floresta aleatória)
ROMMA	*relaxed online maximum margin algorithm* (algoritmo de margem máxima *online* relaxada)
SMS	*short message service* (serviço de mensagens curtas)
SGD	*stochastic gradient descent* (gradiente descendente estocástico)
SPIM	*spam over instant messaging* (spam de mensagens instantâneas)
SVM	*support vector machines* (máquinas de vetores de suporte)
TF	*term frequency* (frequência do termo)

TF-IDF	*term frequency-inverse document frequency* (frequência do termo-frequência inversa dos documentos)
XML	*extensible markup language* (linguagem de marcação extensível)
YSC	YouTube Spam *Collection*

Lista de símbolos

$\mathcal{D}^\star$	conjunto de treinamento
$\lvert\mathcal{D}^\star\rvert$	quantidade de instâncias de treinamento
d_i	i-ésima instância. No contexto de categorização de texto, é o i-ésimo documento de texto
$c(d)$	classe do documento d
$\lvert\mathcal{C}\rvert$	quantidade de classes
c_j	j-ésima classe
$\mathfrak{mc}_{i,j}$	valor da i-ésima linha e da j-ésima coluna da matriz de confusão e que corresponde ao número de exemplos classificados como pertencentes à i-ésima classe, mas que na verdade pertencem à j-ésima classe.
vp_j	verdadeiros positivos relativos à j-ésima classe
fp_j	falsos positivos relativos à j-ésima classe
vn_j	verdadeiros negativos relativos à j-ésima classe
fn_j	falsos negativos relativos à j-ésima classe
$p_\star$	probabilidade total de concordância entre a classe verdadeira e a classe predita pelo classificador
$p_\circ$	probabilidade de concordância ao acaso entre a classe verdadeira e a classe predita pelo classificador
$\mathfrak{mc}_{i,\cdot}$	soma dos elementos da i-ésima linha da matriz de confusão
$\mathfrak{mc}_{\cdot,i}$	soma dos elementos da i-ésima coluna da matriz de confusão
$\mathcal{D}$	conjunto de dados
$\lvert\mathcal{D}\rvert$	quantidade de instâncias
$\mathcal{T}$	conjunto de atributos obtidos após aplicar OHE. No contexto de categorização de texto, é o conjunto de termos
t_i	i-ésimo atributo após aplicar OHE. No contexto de categorização de texto, é o i-ésimo termo
$\lvert\mathcal{T}\rvert$	quantidade de atributos obtidos após aplicar OHE. No contexto de categorização de texto, é a quantidade de termos
$w(t_i, d)$	peso do i-ésimo termo do documento d
$\hat{w}(t_i, d)$	peso do i-ésimo termo do documento d após a normalização L2
$\overline{TF}(t_i, d)$	$\log(1 + TF(t_i, d))$, isto é, a frequência do i-ésimo termo normalizada
$TF(t_i, d)$	frequência do i-ésimo termo
DF_{t_i}	número de documentos de treinamento que contém o i-ésimo termo
X	variável aleatória discreta
$\mathcal{X}$	alfabeto

x_i	i-ésimo elemento do alfabeto $\mathcal{X}$
$H(X)$	entropia de X
$p(x_i)$	probabilidade de x_i
$I(x_i)$	valor da informação de x_i
Z	sequência de símbolos
$\mathcal{C}(x_i)$	código correspondente à x_i
$\mathcal{A}$	alfabeto binário
$\mathcal{A}^*$	todas as sequências que podem ser formadas usando o alfabeto $\mathcal{A}$
$L_{\mathcal{C}}(x_i)$	tamanho de $\mathcal{C}(x_i)$
M_i	i-ésimo modelo
$\mathcal{C}$	conjunto de todas as possíveis classes
$L(d\|c_j)$	tamanho da descrição de d em relação à j-ésima classe
$\hat{S}(d, c_j)$	penalidade dada para a classe c_j
$\|d\|$	quantidade de termos no documento d
$L(t_i\|c_j)$	tamanho da descrição do i-ésimo termo em relação à j-ésima classe
$K(t_i)$	penalidade aplicada ao i-ésimo termo
n	variável que armazena a soma dos pesos de cada termo em $\mathcal{T}$ para cada classe em $\mathcal{C}$
$\hat{n}$	variável que armazena a soma dos pesos em n para cada classe em $\mathcal{C}$
$\|\Omega\|$	porção do tamanho de descrição para termos que nunca apareceram em $\mathcal{D}^\star$
$S(d, \overline{c}_j)$	similaridade de cosseno
$\overline{c}_j$	vetor protótipo da j-ésima classe
$\|\hat{\mathcal{D}}_{c_j}\|$	quantidade de documentos de treinamento pertencentes à j-ésima classe
$\hat{w}(:, d)$	todos os pesos normalizados dos termos do documento d
$\|\hat{\mathcal{D}}\|$	variável que armazena o número de documentos para cada classe em $\mathcal{C}$
μ	constante usada para evitar que o denominador seja 0
$F(t_i)$	pontuação de contribuição do i-ésimo termo
ϕ	frequência de cada termo em $\mathcal{T}$ para cada classe em $\mathcal{C}$
$\|\bar{d}\|$	quantidade média de termos por documento
τ	índice da classe mais frequente
$p(c_j\|t_i)$	probabilidade condicional da j-ésima classe, dada a presença do i-ésimo termo
$p(\bar{t}_i\|c_j)$	probabilidade condicional da ausência do i-ésimo termo, dada a j-ésima classe
$p(t_i\|\bar{c}_j)$	probabilidade condicional do i-ésimo termo
$p(\bar{t}_i\|\bar{c}_j)$	probabilidade condicional da ausência do i-ésimo termo
ψ	mediana de termos por documento
k	quantidade de modelos ou grupos avaliados na análise estatística
q	quantidade de observações consideradas na análise estatística

R_j	média dos *rankings* do j-ésimo modelo ou grupo avaliado na análise estatística
α	intervalo de confiança
$\mathcal{B}$	conjunto de atributos categóricos
$\|\mathcal{B}\|$	quantidade de atributos categóricos
b_i	i-ésimo atributo categórico
$\|b_i\|$	quantidade de possíveis valores do i-ésimo atributo categórico
$v_i(b_j)$	i-ésimo valor do j-ésimo atributo categórico
e	parâmetro que ajusta a velocidade de crescimento da função de penalidade K no método `MDLClass`

Sumário

1 Introdução

Métodos de aprendizado de máquina buscam entre um espaço de potenciais modelos aquele que melhor se ajusta aos dados. Para atingir tal propósito, eles fazem uso do conhecimento prévio obtido pela observação de dados que lhes foram apresentados. Nesse sentido, é desejável obter modelos que se ajustem adequadamente aos dados conhecidos, mas que ao mesmo tempo consigam reproduzir bem o comportamento de dados ainda não observados. Isso é conhecido como capacidade de generalização do modelo.

Uma vez que existe um grande número de modelos candidatos, cada método de aprendizado de máquina adota uma estratégia de seleção, bem como uma forma distinta para representar cada modelo. Uma maneira simples de realizar essa seleção é privilegiar o modelo que oferece a descrição mais compacta dos dados de observação. Neste caso, a estratégia de seleção é baseada no princípio da parcimônia, também conhecido como princípio da navalha de Occam (Occam's *razor*). Esse princípio foi formulado por William de Occam, na Idade Média, como uma crítica à filosofia escolástica que utilizava teorias complexas para explicar a realidade. Hoje em dia, o princípio da navalha de Occam é vastamente usado em metodologia científica, sugerindo que entre duas ou mais hipóteses que possam explicar um mesmo fenômeno, provavelmente a hipótese mais simples seja a mais adequada (DOMINGOS, 1999).

Em Teoria da Informação, uma formalização do princípio da navalha de Occam foi desenvolvida por Rissanen (1978). Ela ficou conhecida como princípio da descrição mais simples (MDL – *minimum description length*), cuja ideia principal é que entre dois ou mais modelos candidatos deve-se escolher aquele que possui o menor tamanho de descrição. Isso significa que modelos menos complexos são preferíveis.

O princípio MDL tem suas raízes na complexidade de Kolmogorov, que é definida como o tamanho do menor programa que descreve um objeto representado por uma sequência binária (KOLMOGOROV, 1965; LI; VITÁNYI, 2008). Quanto menor a complexidade de Kolmogorov de uma determinada sequência, mais regularidade ela apresenta. Portanto, o menor programa que consegue imprimir uma determinada sequência é também o que mais a compreendeu e tem mais conhecimento sobre ela, logo ele deve ser escolhido para representá-la.

O princípio MDL incorpora os fundamentos da navalha de Occam, da complexidade de Kolmogorov e, ao mesmo tempo, procura selecionar o modelo que melhor representa os dados. Consequentemente, ele gera um balanceamento entre a complexidade do modelo e a qualidade do ajuste aos dados de observação. Tais características são bastante desejáveis na área de aprendizado de máquina, pois naturalmente evitam o problema de sobreajustamento (*overfitting*). Uma hipótese altamente complexa, que se

ajusta extremamente bem aos dados observados, normalmente comete muitos erros em dados não conhecidos, ou seja, possui baixa capacidade de generalização.

Os fundamentos teóricos do princípio MDL, bem como suas características, tornam a sua aplicação atraente para a área de aprendizado de máquina, principalmente em problemas complexos e dinâmicos como categorização de textos, que frequentemente possui alta dimensionalidade. Existem evidências de que a aplicação do princípio MDL nesses cenários pode obter resultados promissores (ALMEIDA *et al.*, 2010; ALMEIDA; YAMAKAMI, 2012). Isto deve-se ao fato da sua robustez aos seguintes problemas que afetam a maioria dos métodos de classificação tradicionais:

1. *maldição da dimensionalidade*: o desempenho de alguns métodos de aprendizado tende a ser inferior em problemas de alta dimensão, pois vários desafios surgem em aplicações que envolvem problemas com essa característica. Por exemplo, eles tendem a selecionar modelos mais complexos do que quando o espaço de atributos é de baixa dimensão (HAYKIN, 2008). Modelos mais complexos tendem a usar características particulares dos dados de observação e, portanto, costumam ter baixa capacidade de generalização. O k-vizinhos mais próximo (KNN – *k-nearest neighbours*) e o Bayes ingênuo (NB - *naïve* Bayes) são alguns métodos afetados pela maldição da dimensionalidade (ALMEIDA *et al.*, 2011; FRIEDMAN, 1997).

2. *o alto custo computacional necessário para o treinamento, o que impede a aplicação desses métodos em problemas de aprendizado online*: alguns métodos de classificação, como o método de árvores de classificação e regressão (CART – *classification and regression trees*) (BREIMAN *et al.*, 1984b) e as máquinas de vetores de suporte (SVM – *support vector machines*) (CORTES; VAPNIK, 1995), não podem ser naturalmente treinados de maneira incremental. Portanto, em problemas onde existe a necessidade de atualização constante do modelo de predição, a cada nova amostra, estes métodos devem ser retreinados com todos os exemplos anteriores. O custo computacional do retreinamento, somado à necessidade de manter todos os exemplos de treinamento na memória, inviabilizam o uso desses métodos em problemas de classificação de grande porte.

Idealmente, um método de categorização de texto deveria oferecer um aprendizado rápido, tal como o NB, e robustez contra o problema de sobreajustamento, tal como o SVM. Porém, enquanto em algumas aplicações o desempenho do NB decai quando a dimensionalidade aumenta, o SVM frequentemente apresenta custo computacional de treinamento muito elevado, principalmente se o número de classes do problema é grande e se o espaço de atributos tem alta dimensão (ALMEIDA *et al.*, 2011; WITTEN *et al.*, 2011).

Alguns métodos de aprendizado de máquina tradicionais (*e.g.*, o SVM com funções de *kernel* tradicionais) não podem ser aplicados em problemas de categorização

de texto reais e de grande porte, pois eles requerem que todos os exemplos sejam armazenados na memória ou sejam apresentados simultaneamente, em um processo conhecido como aprendizado *offline* ou aprendizado em batelada. A solução é empregar métodos de aprendizado de máquina que possuem um processo de aprendizado *online*. Em geral, este tipo de método é mais simples e rápido que os métodos *offline*, pois processam um exemplo por vez. Porém, quando os métodos de aprendizado *offline* podem ser aplicados, eles geralmente obtêm melhores desempenhos que os métodos *online* (CRAMMER *et al.*, 2012).

Diante do exposto, esta tese apresenta um método de categorização de textos que emprega seleção de modelo com base no princípio MDL. O método proposto, chamado `MDLText`, possui baixo custo computacional, mesmo para problemas com grande número de documentos e com espaço de atributos de alta dimensionalidade. Ele também pode ser naturalmente aplicado em problemas multiclasse sem a necessidade de técnicas de decomposição de problemas multiclasse em problemas binários, tais como a técnica um-contra-um e variantes (HSU; LIN, 2002). Além disso, o método apresentado pode ser usado em cenários *online* e dinâmicos, pois possui aprendizado incremental.

Objetivos e contribuições

Este manuscrito apresenta um método genérico de categorização de texto baseado no princípio MDL. O método proposto pode ser acoplado à qualquer técnica que seja capaz de determinar uma pontuação aos atributos com base em seu poder discriminativo. Isso torna o método flexível e pode aumentar as chances de sucesso em diferentes aplicações de categorização de texto. Além disso, ele é simples, rápido e pode ser utilizado em problemas reais, dinâmicos e de grande escala, que requerem aprendizado *online*.

Em resumo, este trabalho oferece as contribuições detalhadas a seguir.

- Foi proposto um novo método de categorização de textos baseado no princípio MDL, o `MDLText`, que suporta aprendizado *online* e *offline* e pode ser utilizado em problemas textuais binários e multiclasse.

- O método proposto foi avaliado na classificação de documentos de 45 bases de dados textuais públicas e de grande porte, que representam diferentes aplicações de categorização de texto.

- Foi proposta uma generalização da técnica de fatores de confidência (CF – *confidence factors*) (ASSIS *et al.*, 2006) para que ela possa ser aplicada em problemas multiclasse e ser usada como uma função de pontuação para os termos dos documentos.

- Técnicas geralmente usadas na literatura para a seleção de atributos foram acopladas ao `MDLText` e adaptadas para retornar uma pontuação global para os termos dos documentos.
- O `MDLText` também foi avaliado em um problema de classificação de documentos de texto curtos e ruidosos. Neste problema, o método proposto foi usado em conjunto com técnicas de normalização léxica e indexação semântica. Além disso, foi proposta uma nova abordagem de classificação que combina os termos originais dos documentos com os termos gerados por técnicas de processamento de texto.
- O `MDLText` foi adaptado para ser capaz de processar amostras representadas por qualquer tipo de atributo. O método resultante, chamado `MDLClass`, foi avaliado em experimentos que usaram 23 bases de dados grandes e públicas em um cenário de aprendizado *offline*.

Organização do trabalho

Para facilitar a leitura e a compreensão deste trabalho, ele foi estruturado como descrito a seguir.

- No Capítulo 2, são apresentados conceitos básicos de aprendizado de máquina e são abordados os principais paradigmas de aprendizado. Também, são introduzidos os conceitos de classificação e regressão, além de serem descritos os principais tipos de problemas de classificação. São discutidas também as diferenças entre as abordagens de aprendizado *online* e *offline*. Por fim, são apontadas as principais medidas de desempenho para avaliar métodos de classificação.
- No Capítulo 3, são abordados os conceitos básicos que envolvem a categorização de texto, tais como a representação dos documentos de textos e as técnicas de pré-processamento que podem ser empregadas.
- No Capítulo 4, é feita uma introdução à Teoria da Informação, apresentando conceitos relacionados à medida da informação e da incerteza, além de conceitos sobre codificação e compressão. Posteriormente, é introduzido o princípio MDL e trabalhos que o aplicam no contexto de aprendizado de máquina.
- No Capítulo 5, é apresentado o método de categorização de texto baseado no princípio MDL, o `MDLText`. É descrita a base matemática do método, os algoritmos de treinamento e classificação e o estudo da complexidade computacional.
- No Capítulo 6, são descritas as configurações adotadas nos experimentos realizados para avaliar o `MDLText`. Também, são apresentados os resultados nos cenários de aprendizado *offline* e *online*.

- No Capítulo 7, é introduzida uma proposta inicial de adaptação do `MDLText` para problemas de classificação não-textuais. Após apresentar as bases matemáticas do método gerado por essa adaptação, são descritas as configurações dos experimentos, as bases de dados utilizadas e os resultados obtidos.

- Finalmente, no Capítulo 8, são apresentadas as conclusões e direcionamentos para trabalhos futuros.

2 Aprendizado de máquina

Aprendizado de máquina é uma sub-área da Inteligência Artificial que estuda e desenvolve métodos computacionais que usam a experiência adquirida pela análise e observação de conjuntos de dados para melhorar o seu desempenho e fazer predições (MOHRI *et al.*, 2012; MITCHELL, 1997). Aprendizado de máquina também pode ser definido como uma área de estudo que habilita o computador a aprender sem ser explicitamente programado (SAMUEL, 1959). O conceito de aprendizado está associado a ideia de que o sistema consiga melhorar o seu desempenho em alguma tarefa através da experiência (MITCHELL, 1997).

Essa área de estudo recebe contribuições de várias outras áreas tais como ciência da computação, probabilidade, estatística, otimização e álgebra linear. Alguns exemplos de problemas em que o aprendizado de máquina pode ser aplicado incluem: controle de veículos autônomos (BAGNELL *et al.*, 2010), reconhecimento de escrita (PORWAL *et al.*, 2013), reconhecimento de faces (WEI *et al.*, 2011), reconhecimento de fala (ZHANG *et al.*, 2013), categorização de textos (ROSSI *et al.*, 2016; WU *et al.*, 2014), tradução automática de textos (GONZÁLEZ-RUBIO; CASACUBERTA, 2014), detecção de fraudes (BAHNSEN *et al.*, 2016), diagnóstico de doenças (TSENG *et al.*, 2014), previsão do tempo (HAMIDI *et al.*, 2015) e sistemas de recomendação de filmes, livros e outros produtos (BRBIĆ; ŽARKO, 2015).

Neste capítulo, são oferecidos, de maneira sucinta, conceitos básicos de aprendizado de máquina, tais como definições e exemplos de aprendizado supervisionado e não-supervisionado. Também, são introduzidas as definições de classificação e regressão, além de serem descritos os tipos de problemas de classificação. Por fim, são discutidas as diferenças entre as abordagens de aprendizado *online* e *offline* e são apresentadas as principais medidas de desempenho usadas para avaliar métodos de classificação.

2.1 Seleção de modelos

Métodos de aprendizado de máquina podem ser vistos como métodos de seleção de modelos pois, dado um conjunto de modelos candidatos, eles procuram escolher um que se ajuste bem aos dados. Para isso, eles usam informações extraídas de um conjunto pré-existente de dados, com o objetivo de fazer boas predições em dados futuros.

No processo de seleção de modelos, um dos problemas que podem ser enfrentados pelos métodos de aprendizado é o dilema *bias*-variância: deseja-se selecionar um modelo que se ajuste bem aos dados já observados, mas que ao mesmo tempo seja genérico o suficiente para reproduzir bem o comportamento de dados não observados.

Modelos com alta variância, geralmente são mais complexos e se ajustam muito bem aos dados conhecidos, porém possuem baixa capacidade de generalização e, portanto, cometem muitos erros em dados não conhecidos (sobreajustamento ou *overfitting*). Modelos com alto *bias*, geralmente possuem baixa complexidade e não conseguem capturar regularidades importantes nos dados observados que os ajudem a fazer boas predições em dados futuros (subajustamento ou *underfitting*) (BISHOP, 2006; HASTIE *et al.*, 2009).

Os métodos de aprendizado de máquina podem adotar diferentes estratégias de seleção de modelos. Geralmente, eles fazem uso de parâmetros de ajuste da complexidade do modelo. Por exemplo, métodos de aprendizado que usam funções de *kernel* podem adotar parâmetros que ajustam a largura do *kernel*, métodos baseados em funções polinomiais podem usar parâmetros que ajustam o grau do polinômio e métodos baseados em vizinhança podem usar um parâmetro que ajusta o número de vizinhos que serão analisados. Porém, existem outros métodos que adotam estratégias de seleção de modelos baseadas no princípio da Navalha de Occam. Portanto, eles consideram que, dado um conjunto de modelos candidatos com capacidade de predição similar, o modelo mais simples provavelmente é a melhor escolha (HASTIE *et al.*, 2009).

2.2 Tipos de aprendizado

Os métodos de aprendizado de máquina podem ser classificados de acordo com o processo de construção do seu modelo de predição. Sendo assim, alguns dos principais tipos de problemas de aprendizado de máquina são: aprendizado supervisionado, não-supervisionado, semi-supervisionado e por reforço.

2.2.1 Aprendizado supervisionado

No aprendizado supervisionado, é apresentado ao método de aprendizado um conjunto de dados $\mathcal{D}^\star = \{d_1, d_2, ..., d_{|\mathcal{D}^\star|}\}$, chamado de conjunto de treinamento, onde $d_i \in \mathcal{D}^\star$ possui um atributo-alvo $c(d_i)$ associado, formando um conjunto de pares $(d_i, c(d_i))$ (BISHOP, 2006; MURPHY, 2012). Portanto, a principal característica desse tipo de aprendizado é que se conhece a saída desejada ou esperada (atributo-alvo) para cada dado que é apresentado ao modelo durante seu treinamento. Por exemplo, em um problema de diagnóstico médico, os possíveis valores do atributo-alvo associados a cada dado poderiam ser "paciente com dengue" e "paciente sem dengue". Em um problema de reconhecimento de caracteres, os valores do atributo-alvo poderiam ser as letras do alfabeto. O atributo-alvo também pode assumir valores contínuos, como na predição do valor das ações de uma empresa na bolsa de valores.

Os dados de treinamento, juntamente com seus atributos-alvo, são usados pelos métodos para a construção de uma função ou modelo de predição $\mathcal{F}$ que posteriormente

pode ser utilizado para predizer os atributos-alvo de dados desconhecidos. Depois que o modelo de predição é criado, ele pode ser testado em um outro conjunto de dados, chamado de conjunto de teste.

Os dados de um problema podem ser chamados de exemplos ou instâncias. Eles são representados por um conjunto de atributos (em inglês, *features*). Por exemplo, em um problema de diagnóstico médico, um determinado paciente poderia ser descrito pelos atributos idade, sexo, peso, pressão sanguínea e temperatura corporal.

Os problemas de aprendizado supervisionado podem ser divididos de acordo com os possíveis valores do atributo-alvo. Se o atributo-alvo parte de um conjunto finito com valores discretos, o problema é chamado de classificação e o atributo-alvo é chamado de classe. Se o atributo-alvo pode assumir valor contínuo, o problema é chamado de regressão (MITCHELL, 1997; BISHOP, 2006).

Os problemas de classificação também podem ser divididos em binários, multiclasse, multilabel e hierárquicos.

2.2.1.1 Classificação binária

Na classificação binária, os dados podem ser atribuídos a uma entre duas possíveis classes. Por exemplo, em um problema de classificação de e-mails, as mensagens podem ser classificadas como spam ou não-spam.

2.2.1.2 Classificação multiclasse

Na classificação multiclasse, os dados podem ser atribuídos a uma entre duas ou mais classes. Por exemplo, em um problema de reconhecimento de dígitos cursivos, a classe de um dígito poderia ser qualquer valor entre zero e nove.

Muitos métodos clássicos de classificação não podem ser aplicados em problemas multiclasse diretamente. Porém, existem técnicas que fazem a decomposição do problema em vários problemas binários. Duas técnicas bastantes conhecidas são a um-contra-todos (*one-against-all*) e um-contra-um (*one-against-one*) (HSU; LIN, 2002).

A técnica um-contra-todos constrói $|\mathcal{C}|$ modelos de predição binários, onde $|\mathcal{C}|$ é o número de classes. Cada modelo é treinado para separar uma das classes das demais. Portanto, o i-ésimo modelo é treinado considerando que todos os exemplos da i-ésima classe pertencem à classe positiva, enquanto os exemplos das outras classes pertencem à classe negativa. Quando um exemplo de classe desconhecida é apresentado, ele é classificado por cada um dos $|\mathcal{C}|$ modelos de predição e recebe a classe relativa ao modelo que obtiver o melhor resultado (HSU; LIN, 2002).

Na técnica um-contra-um, dadas $|\mathcal{C}|$ classes, são construídos $\frac{|\mathcal{C}|(|\mathcal{C}|-1)}{2}$ modelos de predição, um para cada possível par de classes. Quando um exemplo de classe desco-

nhecida é apresentado, ele é classificado por cada um dos modelos de predição e recebe a classe que for escolhida pela maioria deles (HSU; LIN, 2002).

2.2.1.3 Classificação *multilabel*

Na classificação *multilabel*, os dados podem ser atribuídos a uma ou várias classes. Por exemplo, em um problema de classificação de notícias, uma amostra poderia ser atribuída à classe de notícias políticas e econômicas.

2.2.1.4 Classificação hierárquica

Na classificação hierárquica, as classes são organizadas em uma hierarquia de classes pré-definidas. Portanto, os dados podem ser classificados como pertencentes à uma classe que é dividida em subclasses ou pertence à superclasses. Por exemplo, em um problema de classificação de notícias, uma amostra pode ser atribuída à classe esportes e depois à subclasse futebol.

2.2.1.5 Métodos de classificação

Existem diversos métodos de classificação disponíveis na literatura, sendo que todos utilizam algum tipo de representação de hipótese e alguma estratégia de seleção de modelo. As mais tradicionais são:

- *métodos baseados em otimização*: por exemplo, as máquinas de vetores de suporte (SVM – *support vector machines*) (CORTES; VAPNIK, 1995) e as redes neurais artificiais (HAYKIN, 1998);
- *métodos baseados em árvores de decisão (DT – decision trees)*: por exemplo, as árvores de classificação e regressão (CART – *classification and regression trees*) (BREIMAN *et al.*, 1984b), o ID3 (QUINLAN, 1986) e o C4.5 (QUINLAN, 1996);
- *métodos probabilísticos*: por exemplo, o Bayes ingênuo (NB - *naïve* Bayes) (MCCALLUM; NIGAM, 1998);
- *métodos baseados em vizinhança*: por exemplo, o k-vizinhos mais próximo (KNN – k-nearest neighbours) (COVER; HART, 1967);
- *métodos que fazem a combinação de diversos modelos de predição (ensemble)*: por exemplo, a floresta aleatória (RF – *random forest*) (BREIMAN, 2001), o *bagging* (BREIMAN, 1996) e o *boosting* adaptativo (FREUND; SCHAPIRE, 1996); e
- *métodos baseados em similaridade*: por exemplo, o Rocchio (ROCCHIO, 1971).

2.2.2 Aprendizado não-supervisionado

No aprendizado não-supervisionado, os dados não são rotulados, ou seja, eles não possuem um atributo-alvo ou os valores do atributo-alvo não são conhecidos. Essa classe de métodos tem como objetivo descrever os dados através do agrupamento das amostras que possuem padrões semelhantes, o que é conhecido como *clusterização*, ou para outros fins tais como descobrir regularidades nos dados ou representar os dados usando um menor número de dimensões (BISHOP, 2006). O mapa auto-organizável de Kohonen (KOHONEN, 1990) e o K-médias (HASTIE *et al.*, 2009) são exemplos de métodos de aprendizado não-supervisionado.

2.2.3 Aprendizado semi-supervisionado

No aprendizado semi-supervisionado, parte dos dados apresentados aos métodos de aprendizado são rotulados e parte deles não possui rótulos. Esse tipo de aprendizado é indicado para problemas em que a obtenção de exemplos rotulados é muito custosa. Com poucos exemplos rotulados, muitas vezes os métodos supervisionados não são capazes de criar um modelo de predição bom o suficiente. Em alguns casos, o uso de aprendizado não-supervisionado também não é indicado, pois pode ser difícil entender o que os grupos encontrados pelo método significam. Nesse tipo de problema, o uso de alguns dados rotulados pode ajudar a identificar o rótulo de cada grupo (AGGARWAL, 2014).

2.2.4 Aprendizado por reforço

No aprendizado por reforço, os agentes aprendem a partir de reforços, como recompensas e punições, de acordo com as decisões tomadas. Nesse tipo de aprendizado, não é fornecida uma resposta precisa ao método, mas apenas a informação se a ação tomada foi correta ou não. Portanto, o aprendizado ocorre pela interação com o ambiente, na forma de tentativa e erro, em que o objetivo do algoritmo ou agente de aprendizagem é aumentar a sua recompensa (BISHOP, 2006). Por exemplo, em um problema de navegação autônoma, o aprendizado pode ocorrer aplicando punições sempre que o agente colidir com um obstáculo e dando recompensas quando o agente atingir um objetivo (SMART; KAELBLING, 2002).

2.3 Aprendizado *online*

Os métodos de aprendizado de máquina podem construir e atualizar o modelo de predição usando dois cenários diferentes: *offline* e *online*.

No aprendizado *offline*, o mais comum na literatura, todos os exemplos de treinamento são apresentados ao método de classificação de uma única vez e este cria um

modelo de predição global para classificar dados não conhecidos. Os modelos gerados no aprendizado *offline* geralmente são mais poderosos, porém são estáticos.

Nos métodos que suportam apenas treinamento *offline*, como o SVM, CART e RF, depois que é feito o treinamento, caso seja necessário atualizar o modelo de predição usando novos dados, é necessário refazer o treinamento com todos os dados usados anteriormente e com os novos dados.

No aprendizado *online*, é apresentado um exemplo por vez e o modelo de predição é atualizado de forma incremental. Portanto, métodos de aprendizado *online* são mais apropriados para problemas de larga escala, pois eles não precisam armazenar todos os exemplos na memória. Eles também são apropriados para problemas dinâmicos que requerem atualização frequente do modelo de predição. Porém, em cenários onde métodos *offline* podem ser aplicados, eles geralmente obtém melhores resultados que os métodos de aprendizado *online* (CRAMMER *et al.*, 2012).

Um cenário tradicional onde o aprendizado *online* pode ser aplicado é na filtragem de spam. Nesse problema, o método é usado em uma aplicação real e faz suas predições usando um modelo previamente treinado. Por exemplo, quando um email é classificado erroneamente como não-spam, indo para a caixa de emails do usuário, este pode avisar ao serviço de email que tal mensagem é um spam. Então, essa informação é fornecida para o método de classificação que usa o rótulo correto para treinar e atualizar o modelo de predição.

Existem vários métodos de aprendizado *online* na literatura, sendo que um dos mais clássicos é o perceptron (ROSENBLATT, 1958). Alguns métodos são baseados no critério de margem máxima, como o algoritmo de margem máxima *online* relaxada (ROMMA – *relaxed online maximum margin algorithm*) (LI; LONG, 2002). Outros são baseados no gradiente descendente, como o gradiente descendente *online* (OGD – *online gradient descent*) (ZINKEVICH, 2003) e o gradiente descendente estocástico (SGD – *stochastic gradient descent*) (ZHANG, 2004). Existem ainda métodos probabilísticos, tais como o NB.

2.4 Medidas de desempenho

Para calcular o desempenho de um método de classificação, é comum empregar medidas baseadas na matriz de confusão. Essa matriz tem tamanho $|\mathcal{C}| \times |\mathcal{C}|$, onde $|\mathcal{C}|$ representa o número de classes do problema. As colunas da matriz representam as classes verdadeiras do problema, enquanto as linhas representam as classes preditas pelo método de classificação. Um exemplo de matriz de confusão é apresentado na Tabela 3, onde $\mathfrak{mc}_{ij}$ é o número de exemplos classificados como pertencentes à classe c_i, mas que na verdade pertencem à classe c_j.

Tabela 3 – Matriz de confusão para problemas multiclasse.

	Classe verdadeira			
	c_1	c_2	...	$c_{\|\mathcal{C}\|}$
Classificado como c_1	$\mathfrak{mc}_{11}$	$\mathfrak{mc}_{12}$	...	$\mathfrak{mc}_{1\|\mathcal{C}\|}$
Classificado como c_2	$\mathfrak{mc}_{21}$	$\mathfrak{mc}_{22}$	...	$\mathfrak{mc}_{2\|\mathcal{C}\|}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Classificado como $c_{\|\mathcal{C}\|}$	$\mathfrak{mc}_{\|\mathcal{C}\|1}$	$\mathfrak{mc}_{\|\mathcal{C}\|2}$	...	$\mathfrak{mc}_{\|\mathcal{C}\|\|\mathcal{C}\|}$

Dada uma classe c_j, usando-se os valores apresentados na matriz de confusão, pode-se calcular os seguintes valores:

- *verdadeiros positivos* $\rightarrow vp_j = \mathfrak{mc}_{jj}$: quantidade de exemplos corretamente classificados como pertencentes à classe c_j;
- *falsos positivos* $\rightarrow fp_j = \sum_{k=1|k\neq j}^{|\mathcal{C}|} \mathfrak{mc}_{jk}$: quantidade de exemplos incorretamente classificados como pertencentes à classe c_j;
- *verdadeiros negativos* $\rightarrow vn_j = \sum_{i=1|i\neq j}^{|\mathcal{C}|} \sum_{k=1|k\neq j}^{|\mathcal{C}|} \mathfrak{mc}_{ik}$: quantidade de exemplos corretamente classificados como não pertencentes à classe c_j;
- *falsos negativos* $\rightarrow fn_j = \sum_{k=1|k\neq j}^{|\mathcal{C}|} \mathfrak{mc}_{kj}$: quantidade de exemplos da classe c_j incorretamente classificados como pertencentes a outra classe.

Uma vez obtida a quantidade de verdadeiros (falsos) positivos, verdadeiros (falsos) negativos, é possível calcular diferentes medidas de desempenho. Algumas delas são definidas apenas para problemas binários, onde existe uma classe alvo que é chamada de classe positiva. A outra é chamada de classe negativa. Por exemplo, em um problema de classificação de spam, a classe positiva é a classe spam e a negativa é a classe *ham* (não-spam). Considerando, por exemplo, que em um problema binário, a classe c_1 é a classe positiva, algumas possíveis medidas de desempenho que podem ser utilizadas para avaliar a qualidade da classificação são apresentadas a seguir (SOKOLOVA; LAPALME, 2009; ALMEIDA *et al.*, 2011; WITTEN *et al.*, 2011).

- *Acurácia*: é uma das medidas mais usadas na literatura. Ela representa a taxa de acerto global do método de classificação, ou seja, a proporção de predições corretas em relação ao tamanho do conjunto de dados. Ela não é aconselhada para problemas que possuem classes desbalanceadas, pois ela é dominada pelo desempenho do classificador na predição das amostras da classe mais frequente. A acurácia pode ser calculada da seguinte forma:

$$\text{acurácia} = \frac{vp_1 + vn_1}{vp_1 + vn_1 + fp_1 + fn_1}. \tag{2.1}$$

- *Sensitividade, recall ou taxa de verdadeiros positivos*: proporção de amostras da classe positiva identificada corretamente. Indica o quão bom o classificador é em amostras da classe positiva. A sensitividade pode ser calculada pela seguinte equação:

$$\text{sensitividade} = \frac{vp_1}{vp_1 + fn_1}. \tag{2.2}$$

- *Especificidade ou taxa de verdadeiros negativos*: proporção de amostras da classe negativa identificada corretamente. Indica o quão bom o classificador é em amostras da classe negativa. A especificidade pode ser calculada da seguinte maneira:

$$\text{especificidade} = \frac{vn_1}{vn_1 + fp_1}. \tag{2.3}$$

- *Taxa de falsos positivos*: proporção de amostras da classe negativa identificada incorretamente. Ela pode ser calculada por:

$$\text{taxa de falsos positivos} = \frac{fp_1}{fp_1 + vn_1}. \tag{2.4}$$

- *Precisão*: porcentagem de amostras classificadas como pertencentes à classe positiva e que realmente pertencem à classe positiva. Ela pode ser calculada da seguinte maneira:

$$\text{precisão} = \frac{vp_1}{vp_1 + fp_1}. \tag{2.5}$$

- *F-medida*: média harmônica entre precisão e sensitividade. Ela pode variar entre zero e um, sendo que valores próximos de um indicam que o classificador obteve bons resultados tanto na precisão, quanto na sensitividade. A F-medida é calculada pela seguinte equação:

$$\text{F-medida} = 2 \times \frac{\text{precisão} \times \text{sensitividade}}{\text{precisão} + \text{sensitividade}}. \tag{2.6}$$

- *Coeficiente de correlação de Matthews (MCC – Matthews correlation coefficient)*: medida amplamente utilizada em problemas de classificação binária que tem como principal característica levar em conta o desbalanceamento entre as classes. O MCC pode ser calculado da seguinte forma:

$$\text{MCC} = \frac{vp_1 \times vn_1 - fp_1 \times fn_1}{\sqrt{(vp_1 + fp_1) \times (vp_1 + fn_1) \times (vn_1 + fp_1) \times (vn_1 + fn_1)}}. \tag{2.7}$$

  O MCC retorna um valor que varia entre -1 e +1. Um valor próximo de +1 indica uma boa predição. Por outro lado, um valor próximo de -1 indica que o classificador

faz predições inversas. Por fim, um MCC igual a zero significa que o método de classificação tem desempenho equivalente a um preditor aleatório (ALMEIDA *et al.*, 2011).

Para problemas binários sem uma classe-alvo ou para problemas multiclasse, normalmente são utilizadas medidas de desempenho que consideram a média entre os resultados relativos à cada classe do problema. As duas principais estratégias para obter a média de desempenho entre as classes são a média macro e a média micro.

Na média macro, o resultado final é a média dos resultados obtidos para cada classe do problema. Além disso, ela considera que todas as classes possuem a mesma importância. Por outro lado, a média micro acumula a quantidade de verdadeiros (falsos) positivos e verdadeiros (falsos) negativos para calcular o desempenho do classificador. Nessa estratégia, o resultado final é dominado pelas classes mais frequentes, o que pode ser um problema quando as classes são muito desbalanceadas. A seguir, são apresentadas algumas medidas de desempenho que usam tais estratégias (SOKOLOVA; LAPALME, 2009):

- macro sensitividade $= \frac{1}{|\mathcal{C}|} \times \sum_{j=1}^{|\mathcal{C}|} \frac{vp_j}{vp_j+fn_j}$;
- macro precisão $= \frac{1}{|\mathcal{C}|} \times \sum_{j=1}^{|\mathcal{C}|} \frac{vp_j}{vp_j+fp_j}$;
- macro F-medida $= 2 \times \frac{\text{macro precisão} \times \text{macro sensitividade}}{\text{macro precisão} + \text{macro sensitividade}}$;
- micro sensitividade $= \frac{\sum_{j=1}^{|\mathcal{C}|} vp_j}{\sum_{j=1}^{|\mathcal{C}|} vp_j+fn_j}$;
- micro precisão $= \frac{\sum_{j=1}^{|\mathcal{C}|} vp_j}{\sum_{j=1}^{|\mathcal{C}|} vp_j+fp_j}$;
- micro F-medida $= 2 \times \frac{\text{micro precisão} \times \text{micro sensitividade}}{\text{micro precisão} + \text{micro sensitividade}}$.

Outra medida de desempenho que pode ser usada para problemas multiclasse é o coeficiente Kappa, que é calculado pela seguinte equação:

$$\text{coeficiente Kappa} = \frac{p_\star - p_\circ}{1 - p_\circ}, \tag{2.8}$$

onde $p_\star = \sum_{i=1}^{|\mathcal{C}|} \frac{\mathfrak{mc}_{i,i}}{vp_i+vn_i+fp_i+fn_i}$ é a probabilidade total de concordância entre a classe verdadeira e a classe predita pelo classificador e $p_\circ = \sum_{i=1}^{|\mathcal{C}|} \frac{\mathfrak{mc}_{i,\cdot}}{vp_i+vn_i+fp_i+fn_i} \times \frac{\mathfrak{mc}_{\cdot,i}}{vp_i+vn_i+fp_i+fn_i}$ é a probabilidade de concordância ao acaso. Nesta equação, $\mathfrak{mc}_{i,\cdot}$ é a soma dos elementos da i-ésima linha da matriz de confusão e $\mathfrak{mc}_{\cdot,i}$ é a soma dos elementos da i-ésima coluna. Quando o coeficiente Kappa é positivo, significa que o classificador é melhor que um classificador aleatório. Se o valor é igual a zero, significa que o classificador é equivalente a um classificador aleatório. Se o valor é negativo, o classificador é inferior a um classificador aleatório (BEN-DAVID, 2008).

2.5 Considerações finais

Neste capítulo, foram introduzidos os conceitos básicos da área de aprendizado de máquina, incluindo os paradigmas de aprendizado. Foi dada ênfase ao aprendizado supervisionado, pois este é o tipo de aprendizado abordado neste trabalho.

Foram mostradas também as possíveis categorias de aprendizado supervisionado, que são classificação e regressão. Nesta tese, os problemas abordados são problemas de classificação, mais especificamente problemas binários e multiclasse. Além disso, são consideradas duas dinâmicas de aprendizado: *offline* e *online*. Portanto, neste capítulo, foram apresentadas as duas formas de aprendizado e alguns dos métodos tradicionais de cada uma delas.

Por fim, também foram apresentadas medidas de desempenho tradicionais que podem ser usadas para avaliar a qualidade dos métodos de classificação. Neste estudo, a principal medida empregada na análise dos resultados foi a macro F-medida, pois ela é uma das mais empregadas em problemas multiclasse, combina as medidas precisão e sensitividade e, ainda, a média macro trata todas as classes igualmente, o que é útil em problemas onde as classes são desbalanceadas (SOKOLOVA; LAPALME, 2009; MANNING *et al.*, 2009; YANG, 1999; LIU *et al.*, 2009; ESCALANTE *et al.*, 2015; ZHANG; ZHONG, 2016).

3 Categorização de textos

O volume de informações digitais armazenadas na forma textual vem crescendo desenfreadamente. Atualmente, livros, jornais de notícias, revistas, artigos científicos, entre outros, são geralmente publicados em meio digital, fato que não ocorria há alguns anos. Até mesmo documentos antigos estão sendo digitalizados com o intuito de facilitar o acesso e controle. Somado a isso, existem outros inúmeros serviços, tais como *e-mail*, mensagens de texto compartilhadas através de celulares e redes sociais e comentários em sites e blogs, que aumentam de forma significativa a quantidade de informação textual que precisa ser analisada com o intuito de prover acesso, segurança, organização e demais tipos de ações que colaborem positivamente para a experiência do usuário.

Quando se tem uma grande quantidade de documentos disponíveis, é necessário agrupá-los em categorias para facilitar a análise e extração de conhecimento, bem como a busca e o manuseio. Os humanos têm uma grande facilidade para realizar esse trabalho. Ao ler uma notícia, um livro ou um artigo científico, uma pessoa normalmente consegue saber que tipo de assunto é abordado no documento e associá-lo a um grupo adequado. Porém, quando o número de documentos é muito grande, a categorização manual se torna inviável, pois requer muito tempo e uma quantidade muito grande de mão-de-obra.

Uma das formas de automatizar o processo de categorização de documentos é por meio de métodos de classificação. Esse tipo de método recebe um documento e o classifica em alguma categoria predefinida. Algumas aplicações que vêm empregando métodos de classificação de texto: análise de sentimentos (MUHAMMAD *et al.*, 2016; KATZ *et al.*, 2015), classificação de notícias (GUTLEIN *et al.*, 2009; ROSSI *et al.*, 2016; APHINYANAPHONGS *et al.*, 2014; WU *et al.*, 2014; JIANG *et al.*, 2016), documentos científicos (ROSSI *et al.*, 2012; APHINYANAPHONGS *et al.*, 2014; JIANG *et al.*, 2016; VO; OCK, 2015), emails (UYSAL; GUNAL, 2012; ROSSI *et al.*, 2016; YU; XU, 2008; ALMEIDA *et al.*, 2011; KUMAR *et al.*, 2016) e páginas Web (ROSSI *et al.*, 2012; APHINYANAPHONGS *et al.*, 2014). Recentemente, algumas pesquisas também têm utilizado métodos de classificação em aplicações que envolvem a comunicação através de mensagens curtas, tais como detecção de spam em serviços de mensagens curtas (SMS – *short message service*) (UYSAL *et al.*, 2012; AHMED *et al.*, 2015; GOSWAMI *et al.*, 2016; ALMEIDA *et al.*, 2016), blog spam (CHAN *et al.*, 2015; ALSALEH *et al.*, 2015), problemas relacionados às redes sociais (ALBERTO *et al.*, 2015) e detecção de avaliações *online* fraudulentas (CHAN *et al.*, 2015).

Neste capítulo, são apresentados os conceitos básicos que envolvem a categorização de texto, tais como a forma de representação dos documentos de textos e as técnicas de pré-processamento tradicionalmente empregadas.

3.1 Preparação do texto

Os métodos de aprendizado de máquina geralmente não conseguem processar documentos de texto diretamente, pois estes não possuem uma forma sistematizada. Portanto, é necessário criar uma representação estruturada, como por exemplo, apresentando-os como um conjunto de atributos (SEBASTIANI, 2002; WEISS *et al.*, 2004).

Antes dessa etapa é necessário fazer a coleta dos documentos. Se, por exemplo, um documento de texto precisa ser extraído de uma página Web, é necessário encontrar a parte de interesse do texto no código HTML da página. Por exemplo, em um problema de classificação de notícias, a parte de interesse do texto que pode ajudar a identificar a categoria à qual a notícia pertence é o título e o conteúdo da mesma. Portanto, deve-se procurar as marcações (*tags*) HTML que identificam onde essas partes de interesse estão inseridas. Além disso, as marcações HTML que estão inseridas junto ao conteúdo, como por exemplo as que definem estilos de formatação do texto (negrito, itálico, etc), podem ser descartadas. Esse processo é conhecido como *parsing*. Alguns documentos podem ainda estar armazenados no formato de documento portátil (PDF – *portable document format*), linguagem de marcação extensível (XML – *extensible markup language*), valores separados por vírgulas (CSV –*comma separated values*), entre outros.

Na Figura 1 é apresentado um exemplo de documento de texto da base de dados Reuters-21578[1]. Nesse documento, as informações relevantes para serem usadas no processo de treinamento de um método de classificação são: (1) a categoria do documento que está sendo informada pela *tag* `<D>` que está dentro da *tag* `<TOPICS>`, (2) o título que está informado pela *tag* `<TITLE>` que está dentro da *tag* `<TEXT>` e (3) o conteúdo do texto que está informado pela *tag* `<BODY>` que está dentro da *tag* `<TEXT>`. Para extrair as informações de dentro das *tags* citadas, precisa-se usar um analisador sintático (*parser*).

Geralmente, a etapa de coleta é uma tarefa onerosa, pois cada forma de armazenamento requer uma maneira diferente de extração das informações desejadas para que posteriormente sejam estruturadas de forma a viabilizar seu uso por parte dos métodos de classificação. Ainda, é difícil fazer a limpeza dos documentos para eliminar as informações indesejadas, resolver problemas de codificação de caracteres e garantir a qualidade do texto extraído. Muitas vezes a integridade do texto é comprometida no seu próprio armazenamento por falha humana ou por outros fatores (WEISS *et al.*, 2004; MANNING *et al.*, 2009).

3.1.1 *Tokenização*

Depois que os documentos são coletados, uma etapa importante na estruturação da representação deles é a *tokenização*. Nesta etapa, é feita a divisão do texto em termos (também chamados de *tokens*). Geralmente, um termo é um conjunto de caracteres

[1] A base de dados Reuters-21578 está disponível em ⟨http://goo.gl/mJJTAI⟩. Acessado em: 19/01/2017.

```
<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID="17858" NEWID="
    1440">
<DATE> 4-MAR-1987 09:51:38.24</DATE>
<TOPICS><D>sugar</D></TOPICS>
<PLACES><D>uk</D><D>netherlands</D><D>denmark</D><D>west-germany</D></PLACES>
<PEOPLE></PEOPLE>
<ORGS><D>ec</D></ORGS>
<EXCHANGES></EXCHANGES>
<COMPANIES></COMPANIES>
<UNKNOWN>
&#5;&#5;&#5;C T
&#22;&#22;&#1;f0234&#31;reute
b f BC-U.K.-INTERVENTION-BOA 03-04 0071</UNKNOWN>
<TEXT>&#2;
<TITLE>U.K. INTERVENTION BOARD DETAILS EC SUGAR SALES</TITLE>
<DATELINE> LONDON, March 4 - </DATELINE><BODY>A total 60,500 tonnes of current series
white sugar received export rebates of a maximum 43.147
European Currency Units (Ecus) per 100 kilos at today's
European Community (EC) tender, the U.K. Intervention Board
said.
Out of this, traders in the U.K. Received 43,500 tonnes, in
the Netherlands 12,000, in Denmark 4,000 and in West Germany
1,000 tonnes.
Earlier today, London and Paris traders said they expected
the subsidy for the current season whites campaign for licences
to end-July to be between 43.00 and 43.45 Ecus per 100 kilos.
They had also forecast today's total authorised sugar
tonnage export awards to be between 60,000 and 80,000 tonnes
versus 103,000 last week when the restitution was 43.699 Ecus.
REUTER
&#3;</BODY></TEXT>
</REUTERS>
```

Figura 1 – Exemplo de um documento de texto da base de dados Reuters-21578.

alfanuméricos delimitado por caracteres delimitadores ("\n", "\t", "\r", entre outros), por espaços em branco ou por caracteres não alfanuméricos (ponto, vírgula, asterisco, entre outros). Nesse processo, pode ser definido um limite mínimo e máximo de caracteres para cada termo (MANNING *et al.*, 2009; UYSAL; GUNAL, 2014). Por exemplo, a *tokenização* do texto "*'Esse#restaurante é muito bom! :)*", onde qualquer caractere não-alfanumérico fosse usado como delimitador e o limite minimo de caracteres fosse um, identificaria os seguintes termos: "*Esse*", "*restaurante*", "*muito*" "*bom*".

3.1.2 Técnicas de pré-processamento

Depois da coleta dos documentos, também é necessário tratar os dados textuais para auxiliar o processo de treinamento ou categorização dos documentos. Nesta etapa, frequentemente, são aplicadas algumas técnicas de pré-processamento, conforme apresentado a seguir.

- *Conversão dos termos para letras minúsculas*: evita que duas palavras idênticas sejam consideradas diferentes devido suas letras maiúsculas ou minúsculas (UYSAL; GUNAL, 2014).

- *Remoção de stopwords*: *stopwords* são termos muito comuns, tais como preposições, conjunções e artigos, que fornecem pouca ou nenhuma informação (WEISS *et al.*, 2004). Portanto, é aconselhado que tais termos sejam removidos antes dos processos de treinamento e de classificação.

- *Remoção de palavras raras*: remove os termos com baixa frequência no conjunto de documentos. Termos que aparecem raríssimas vezes no conjunto de treinamento, geralmente não contribuem na identificação da categoria do documento e, portanto, podem ser descartados (WEISS *et al.*, 2004).

- *Estemização*: reduz o termo ao seu radical (UYSAL; GUNAL, 2014). O processo de *estemização* varia de acordo com o idioma. Por exemplo, o algoritmo de *estemização* mais usado para documentos da língua inglesa é o algoritmo de Porter (1980). Por outro lado, para a língua portuguesa, o algoritmo de *estemização* mais utilizado na literatura é o RSLP que foi proposto por Orengo e Huyck (2001). Para obter os radicais dos termos, os algoritmos de *estemização* geralmente removem os afixos e o final das palavras (mesmo que não sejam afixos) (MANNING *et al.*, 2009). Alguns exemplos de *estemização* são apresentados na Tabela 4.

- *Lematização*: reduz o termo à sua forma canônica, também conhecida como lema. Para obter o lema de uma palavra, é feita a análise morfológica da mesma (MANNING *et al.*, 2009). A lematização é mais complexa que a *estemização*, pois ela considera a semântica do termo que está sendo analisado e, consequentemente, ela também possui maior custo computacional. Alguns exemplos de lematização são apresentados na Tabela 4.

Tabela 4 – Exemplos de *estemização* e lematização de palavras da língua inglesa e portuguesa.

Palavra Original	***Estemização***	**Lematização**
Palavras da língua inglesa		
flies	*fli*	*fly*
sensational	*sensat*	*sensational*
plotted	*plot*	*plot*
Palavras da língua portuguesa		
felicíssimo	felic	feliz
segurados	segur	segurar
anoitecendo	anoitec	anoitecer

- *Normalização léxica*: consiste em traduzir gírias, expressões idiomáticas e abreviações de palavras, geralmente usadas pelos usuários da Internet, para a sua forma canônica. Para fazer a tradução dos termos são utilizados dicionários conhecidos como *Lingo*, como por exemplo o dicionário `NoSlang`[2] (ALMEIDA *et al.*, 2016). Problemas de categorização de texto que envolvem redes sociais, mensagem de texto ou outros conteúdos gerados por usuários da Web são alguns dos exemplos que requerem o uso da técnica de normalização léxica (ALMEIDA *et al.*, 2016; CLARK; ARAKI, 2011). Um exemplo é apresentado na Tabela 5.

- *Indexação semântica*: esta é uma técnica de extração de informações semânticas que obtém diferentes significados para um determinado termo. O uso dessa técnica tende a aumentar a quantidade de termos de um texto e por isso ela é considerada uma técnica de geração de conceitos, expansão ou enriquecimento textual (ALMEIDA *et al.*, 2016). Para a geração de conceitos, geralmente é utilizado algum repositório semântico, como por exemplo o `LDB BabelNet`[3]. Porém, outras fontes de informação podem ser utilizadas, como por exemplo a Wikipedia (ALMEIDA *et al.*, 2016; VO; OCK, 2015). Na Tabela 5, é apresentado um exemplo de um texto após a aplicação da técnica de indexação semântica.

- *Desambiguação*: técnica também conhecida como *word sense disambiguation*. Ela é empregada para selecionar os conceitos mais relevantes de um termo de acordo com o contexto do texto. Em algumas aplicações, tais como a proposta por Almeida *et al.* (2016), a desambiguação é considerada uma etapa posterior à indexação semântica, pois primeiramente são encontrados diferentes significados para o termo e depois aqueles mais relevantes são selecionados. Um exemplo de desambiguação é apresentado na Tabela 5.

Tabela 5 – Exemplo de normalização léxica, indexação semântica e desambiguação. As técnicas de indexação semântica e desambiguação foram aplicadas após a normalização do texto.

Texto original	*plz there is sth u shud knw abt ur bf.*
Normalização léxica	*please there be something you should know about your boyfriend*
Indexação semântica	*please there be something you should know knowledge noesis about your boyfriend fellow swain young_man*
Desambiguação	*please there be something you should cognition about your boyfriend*

[2] O dicionário `NoSlang` está disponível em: ⟨http://www.noslang.com/dictionary/full/⟩. Acessado em: 19/01/2017.

[3] O `LDB BabelNet` está disponível em: ⟨http://babelnet.org/⟩. Acessado em: 19/01/2017.

Algumas técnicas de pré-processamento podem ser aplicadas após a *tokenização*. Porém, outras, tais como a normalização léxica, indexação semântica e desambiguação devem ser aplicadas antes.

3.1.3 Representação

Após a coleta dos documentos, *tokenização* e pré-processamento, deve-se empregar uma forma mais estruturada de representação para que seja possível usar a maioria dos métodos de classificação. Os documentos podem ser estruturados usando representações em redes, grafos e baseadas em técnicas de redes neurais, como a `Word2Vec` (MIKOLOV *et al.*, 2013; ROSSI *et al.*, 2016; PAPADAKIS *et al.*, 2016; TURIAN *et al.*, 2010). Porém, a forma de representação mais tradicionalmente empregada é o modelo espaço-vetorial, que além de ser simples, obtém bons resultados na maioria das aplicações (SEBASTIANI, 2002). Nesse modelo, cada documento é considerado uma instância ou exemplo do problema de classificação e cada termo representa um possível atributo (WEISS *et al.*, 2004).

No modelo espaço-vetorial, cada termo pode representar uma ou mais palavras do texto. Quando cada um deles representa um par de palavras do texto, eles são chamados de bigramas. Quando eles representam um única palavra, são chamados de unigramas. A representação do documento que usa unigramas e é independente da sequência de palavras, é frequentemente chamada de *bag-of-words* (SEBASTIANI, 2002; AGGARWAL, 2014).

Formalmente, seja $\mathcal{D} = \{d_1, d_2, ..., d_{|\mathcal{D}|}\}$ um conjunto de $|\mathcal{D}|$ documentos e $\mathcal{T} = \{t_1, t_2, ..., t_{|\mathcal{T}|}\}$ o conjunto de $|\mathcal{T}|$ termos extraídos dos documentos desse conjunto. No modelo espaço vetorial, cada documento pode ser representado por uma matriz documento-termo, onde cada posição dela apresenta um peso $w(t_i, d_j)$ associado a cada termo, conforme mostra a Tabela 6.

Tabela 6 – Representação dos documentos usando o modelo espaço-vetorial.

Documentos	Termos				
	t_1	t_2	t_3	...	$t_{\|\mathcal{T}\|}$
d_1	$w(t_1, d_1)$	$w(t_2, d_1)$	$w(t_3, d_1)$	...	$w(t_{\|\mathcal{T}\|}, d_1)$
d_2	$w(t_1, d_2)$	$w(t_2, d_2)$	$w(t_3, d_2)$	...	$w(t_{\|\mathcal{T}\|}, d_2)$
d_3	$w(t_1, d_3)$	$w(t_2, d_3)$	$w(t_3, d_3)$	...	$w(t_{\|\mathcal{T}\|}, d_3)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$d_{\|\mathcal{D}\|}$	$w(t_1, d_{\|\mathcal{D}\|})$	$w(t_2, d_{\|\mathcal{D}\|})$	$w(t_3, d_{\|\mathcal{D}\|})$	...	$w(t_{\|\mathcal{T}\|}, d_{\|\mathcal{D}\|})$

Os valores usados para representar os termos do documento geralmente são baseados na frequência de ocorrência deles e podem variar dependendo da estratégia de atribuição de pesos que é aplicada, sendo que as mais comuns na literatura são apresentadas a seguir.

- *Frequência do termo (TF – term frequency)*: nesta representação, os pesos dos termos correspondem à quantidade de vezes que o termo apareceu no documento.

- *Frequência do termo-frequência inversa dos documentos (TF-IDF – term frequency-inverse document frequency)*: nesta representação, proposta por Salton *et al.* (1975), o peso do termo é igual ao produto da sua frequência pela frequência inversa dos documentos (IDF – *inverse document frequency*). A IDF é uma medida de quanta informação um termo possui, o que depende do quão frequente ele é no conjunto de dados de treinamento. Um alto valor de TF-IDF é obtido quando o termo tem uma alta frequência no documento que está sendo avaliado e uma baixa frequência no conjunto de dados de treinamento (WILBUR; KIM, 2009; RENNIE *et al.*, 2003). Para calcular o peso TF-IDF de um termo, pode-se aplicar a seguinte equação:

$$\begin{aligned} w(t_i, d) &= \overline{TF}(t_i, d) \times IDF_{t_i} \\ &= \log(1 + TF(t_i, d)) \times \log\left(\frac{|\mathcal{D}^\star| + 1}{DF_{t_i} + 1}\right), \end{aligned} \tag{3.1}$$

  onde $TF(t_i, d)$ é a frequência do termo t_i no documento d, $|\mathcal{D}^\star|$ é a quantidade de documentos no conjunto de treinamento e DF_{t_i} é o número de documentos de treinamento que contém o termo t_i (WILBUR; KIM, 2009). A normalização L2 (também conhecida como normalização euclidiana) pode ser aplicada no peso TF-IDF obtido na equação 3.1 (RENNIE *et al.*, 2003):

$$\hat{w}(t_i, d) = \frac{w(t_i, d)}{\|w(:, d)\|_2}. \tag{3.2}$$

- *Binária*: neste tipo de representação, o termo recebe o valor um, caso apareça no documento, e zero, caso não apareça. Portanto, a frequência de vezes que o termo aparece no documento não é considerada. Matematicamente, o peso binário pode ser calculado por:

$$w(t_i, d) = \begin{cases} 1, & \text{se } t_i \text{ aparece em } d\text{;} \\ 0, & \text{se } t_i \text{ não aparece em } d\text{.} \end{cases} \tag{3.3}$$

Depois que o modelo espaço-vetorial é criado, ele pode ser apresentado aos métodos de classificação.

Assim como em qualquer tarefa de classificação, primeiramente deve ser feito o processo de treinamento, onde um conjunto de documentos com suas respectivas classes é apresentado ao método de classificação para a criação do modelo de predição. Depois, documentos sem rótulo são usados para avaliar seu desempenho.

3.1.4 Seleção de atributos

Outra etapa bastante utilizada no processo de categorização de textos é a seleção de atributos. Essa etapa tem como objetivo eliminar atributos pouco informativos, redundantes ou até mesmo ruidosos. Em alguns cenários, ela pode melhorar o desempenho da classificação, aumentando a capacidade de generalização, diminuindo o tempo de aprendizado e simplificando o modelo de predição (BOLÓN-CANEDO *et al.*, 2015; CHANDRASHEKAR; SAHIN, 2014).

As três principais categorias de métodos de seleção de atributos são: filtros, métodos *wrappers* e métodos *embedded*.

- *Filtros*: eles fazem a seleção de atributos antes do processo de treinamento e são independentes do método de classificação. Geralmente, os filtros utilizam informações estatísticas do conjunto de dados de treinamento para determinar a relevância dos atributos. Depois disso, pode ser selecionado o conjunto dos k atributos mais informativos, enquanto os outros são descartados. Outra alternativa, é selecionar apenas os atributos que possuem um valor de relevância maior do que um determinado limiar (LIU; MOTODA, 1998; CHANDRASHEKAR; SAHIN, 2014).

- *Métodos wrappers*: eles utilizam informações sobre o desempenho do método de aprendizado como parte de sua função de avaliação da relevância dos atributos. Eles criam diferentes subconjuntos de atributos, avaliam o desempenho do método de aprendizado usando cada um dos subconjuntos e, geralmente, selecionam aquele que melhora os resultados. Os métodos *wrappers* podem utilizar estratégias de busca para explorar o espaço de todos os possíveis subconjuntos de atributos, pois normalmente a quantidade de combinações é muito extensa, o que inviabiliza uma busca exaustiva (KOHAVI; JOHN, 1997; LIU; MOTODA, 1998; CHANDRASHEKAR; SAHIN, 2014).

- *Métodos embedded*: eles aplicam a seleção de atributos como parte do modelo de predição e são específicos para os métodos de aprendizado ao qual são embutidos. Por exemplo, um método *embedded* pode ser embutido ao modelo de predição como um parâmetro de regularização que atribui pesos aos atributos. Nesse caso, os atributos irrelevantes ou redundantes podem receber um peso zero para que não exerçam nenhuma influência na predição (CHANDRASHEKAR; SAHIN, 2014).

Diversos métodos de seleção de atributos são descritos em Liu e Motoda (1998), Sebastiani (2002), Chandrashekar e Sahin (2014) e Bolón-Canedo *et al.* (2015). Os métodos de seleção baseados em filtros são os mais simples e os mais escaláveis. Já os métodos *wrappers* e *embedded* geralmente demandam um alto custo computacional e por isso, não são apropriados para problemas de categorização de texto reais e de grande escala (LAMIREL *et al.*, 2015; KUMAR; MINZ, 2014).

A maioria dos métodos de seleção de atributos são usados em cenários de aprendizado *offline*. A utilização de métodos de seleção de atributos para cenários de aprendizado *online* ainda é um desafio, pois esse processo requer um número suficiente de exemplos para obter um desempenho estável e aceitável. Além disso, os poucos métodos de seleção de atributos *online* disponíveis na literatura, geralmente consideram que o número de exemplos de treinamento é constante. Ainda, em problemas reais e *online*, sempre que um novo exemplo é apresentado para um método de aprendizado, é necessário considerar a possibilidade de (1) incluir um atributo que foi descartado anteriormente, (2) selecionar novos atributos que estão presentes no novo exemplo e que não haviam ocorrido em nenhum exemplo anterior e (3) remover atributos que foram selecionados anteriormente, mas que depois perderam a relevância (WU *et al.*, 2010; WU *et al.*, 2013; BOLÓN-CANEDO *et al.*, 2015; TANG *et al.*, 2014).

3.2 Considerações finais

Neste capítulo, foram introduzidas brevemente as etapas do processo de categorização e representação de textos. Foi discutido o processo de coleta de documentos e de *tokenização* e foram apresentadas as principais técnicas de pré-processamento que podem facilitar o processo de categorização. Na maioria dos experimentos realizados nesta tese, as técnicas de pré-processamento empregadas foram (1) conversão dos termos para letras minúsculas, (2) remoção de *stopwords* e (3) estemização. Além disso, em um cenário de classificação de documentos curtos e ruidosos, apresentado no Apêndice A, as técnicas de normalização léxica, indexação semântica e desambiguação foram aplicadas para minimizar os problemas de representação e, consequentemente, auxiliar os métodos de classificação.

Também foi discutido neste capítulo como é construído o modelo espaço-vetorial para a representação estruturada dos documentos e foram apresentadas as principais estratégias de atribuição de pesos para os termos de cada documento: TF, TF-IDF e binária. Como o desempenho de alguns métodos de categorização de texto pode ser bastante afetado pela estratégia escolhida, nos experimentos descritos nesta tese, foi realizada uma busca em grade (*grid search*) para selecionar a melhor das três.

Por fim, foram introduzidos os principais tipos de métodos de seleção de atributos e discutidos alguns desafios da aplicação de tais métodos em cenários de aprendizado *online*. Conforme mencionado, a seleção de atributos pode melhorar o desempenho dos métodos de classificação. Portanto, para avaliar o impacto de tal estratégia no método proposto nesta tese, foram realizados experimentos após empregar um método baseado em filtro para reduzir a dimensionalidade dos problemas de categorização de texto. Esse tipo de método foi escolhido em detrimento de métodos *wrappers* ou *embedded*, pois conforme mencionado, os últimos possuem maior custo computacional e são dependentes do

método de classificação.

4 Princípio da descrição mais simples

Problemas de classificação podem ser vistos como problemas de seleção de modelos. Um modelo pode ser criado para cada classe do problema e cada exemplo a ser classificado pode ser representado por um dos modelos disponíveis. Essa forma de seleção de modelos é frequentemente usada por métodos de classificação probabilísticos (BOUCHARD; CELEUX, 2006).

Problemas de seleção de modelos são complexos devido ao dilema *bias*-variância: modelos muito simples falham em capturar regularidades importantes dos dados, enquanto modelos muito complexos tendem a se ajustar a características particulares dos dados conhecidos, que geralmente incluem ruídos, o que gera predições inconsistentes quando amostras desconhecidas são apresentadas. Portanto, são desejáveis modelos com baixa complexidade, mas que ao mesmo tempo se ajustem bem aos dados (BOUCHARD; CELEUX, 2006; HASTIE *et al.*, 2009).

Uma das estratégias de seleção de modelos mais simples é a baseada na Navalha de Occam. Nessa estratégia, modelos menos complexos são priorizados. O princípio MDL, introduzido por Rissanen (1978), é um dos principais métodos de formalização da Navalha de Occam. Ele foi proposto para o problema de seleção de modelos com a prerrogativa de que o melhor modelo para explicar um conjunto de dados é aquele que oferece uma descrição mais compacta dos dados. A descrição mais compacta é oferecida pelo modelo que mais captura regularidade nos dados, ou seja, por aquele que melhor "conhece" os dados de observação (GRÜNWALD *et al.*, 2005). A preferência do princípio MDL por modelos menos complexos pode ajudar a evitar o conhecido problema de sobreajustamento, o que o torna atrativo para problemas de aprendizado supervisionado.

Neste capítulo, é oferecida uma introdução aos principais conceitos relacionados à Teoria da Informação, pertinentes à compreensão do princípio MDL. Assim, são apresentados conceitos relacionados à medida da informação e da incerteza, além de conceitos sobre codificação e compressão. Posteriormente, é apresentado o princípio MDL e são descritos alguns trabalhos que o aplicam na resolução de problemas de aprendizado de máquina.

4.1 Conceitos básicos

A Teoria da Informação é uma área da matemática aplicada que teve um grande desenvolvimento a partir das contribuições do matemático Claude Shannon (COVER; THOMAS, 2006; GRAY, 2011). Essa área de estudo tem influenciado várias outras áreas da ciência, incluindo a área de aprendizado de máquina (MACKAY, 2005; PRIN-

CIPE, 2010).

Um conceito fundamental da Teoria da Informação é o de entropia. Segundo Cover e Thomas (2006), entropia é a medida de incerteza de uma variável aleatória. Seja uma variável aleatória discreta X, que pode assumir um dos valores do alfabeto $\mathcal{X} = \{x_1, x_2, ..., x_{|\mathcal{X}|}\}$ e que possui uma função de massa de probabilidade $p(x_i)$. Sua entropia pode ser calculada por (SHANNON, 1948; COVER; THOMAS, 2006):

$$\begin{aligned} H(X) &= \sum_{i=1}^{|\mathcal{X}|} p(x_i) \times I(x_i) \\ &= -\sum_{i=1}^{|\mathcal{X}|} p(x_i) \times \log p(x_i) \end{aligned} \tag{4.1}$$

Na Equação 4.1, $I(x_i)$ é o valor da informação e está associado à sua relevância. Quanto menor a probabilidade de ocorrência de um evento, maior será sua informação. Formalmente, $I(x_i) = -log(p(x_i))$ (COVER; HART, 1967).

Caso a variável aleatória possa assumir apenas dois valores ($\mathcal{X} = \{x_1, x_2\}$), sendo que a probabilidade de $X = x_1$ é $p(x_1)$ e a probabilidade de $X = x_2$ é $1 - p(x_1)$, a entropia pode ser calculada da seguinte forma:

$$H(X) = -p(x_1) \times \log p(x_1) - [1 - p(x_1)] \times log(1 - p(x_1)). \tag{4.2}$$

Nas Equações 4.1 e 4.2, quando o logaritmo está na base 2, o valor da entropia representa a medida de incerteza em *bits*. Quando é usado o logaritmo natural, é dito que a medida é dada em *nats* (COVER; THOMAS, 2006).

A Figura 2 apresenta o gráfico da função $H(X)$ quando a variável aleatória X pode assumir apenas dois possíveis valores. Neste caso, a entropia atinge o maior valor (em *bits*) quando $p(x_1)$ é igual a 0,5, o que indica máxima incerteza. Além disso, quando $p(x_1)$ é igual a zero ou um, a entropia é igual a zero, o que significa que não existe incerteza.

4.1.1 Código e compressão

O princípio MDL está associado à ideia de compressão de dados. A ideia fundamental é que quanto mais se consegue comprimir um dado, mais conhecimento se tem sobre ele. Supondo que o dado seja uma sequência de símbolos, pode-se definir compressão como o ato de reduzir a quantidade de *bits* necessários para representar essa sequência. Pode-se dizer também que a compressão reproduz a informação contida em uma sequência de símbolos usando uma codificação mais compacta.

Codificação é a ação de descrever uma sequência de símbolos de algum alfabeto por uma outra sequência de símbolos de outro alfabeto. Essa nova representação é denominada código, enquanto o alfabeto é um conjunto finito e contável, cujos elementos são chamados de símbolos (COVER; THOMAS, 2006).

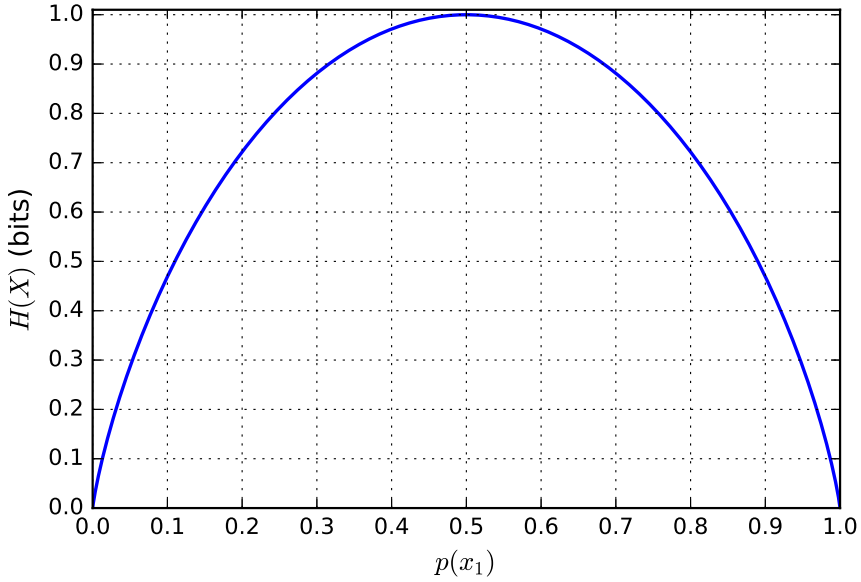


Figura 2 – Entropia de uma variável aleatória X, cujo alfabeto possui apenas dois valores ($\mathcal{X} = \{x_1, x_2\}$).

Suponha que deseje-se transmitir uma determinada sequência Z, cujos símbolos sejam tomados de um alfabeto $\mathcal{X} = \{x_1, x_2, ..., x_{|\mathcal{X}|}\}$, usando uma representação binária, cujo alfabeto é $\mathcal{A} = \{0,1\}$. Para atingir esse objetivo, é necessário mapear cada símbolo do alfabeto $\mathcal{X}$ para uma sequência binária finita e única, que pode ser chamada de palavra-código. Formalmente, o código que mapeia $\mathcal{X}$ para uma representação binária $\mathcal{A}^*$ pode ser definido como uma função $\mathcal{C} : \mathcal{X} \to \mathcal{A}^*$, onde $\mathcal{A}^*$ representa todas as sequências que podem ser formadas usando o alfabeto $\mathcal{A}$. Além disso, $\mathcal{C}(x_i)$ é a palavra-código correspondente à x_i e $L_{\mathcal{C}}(x_i)$ é o tamanho de $\mathcal{C}(x_i)$. Considerando que a probabilidade de ocorrência de cada símbolo $x_i \in \mathcal{X}$ é $p(x_i)$, o tamanho de código esperado para codificar os símbolos do alfabeto $\mathcal{X}$ é:

$$L(\mathcal{C}) = \sum_{i=1}^{|\mathcal{X}|} p(x_i) \times L_{\mathcal{C}}(x_i). \tag{4.3}$$

Por exemplo, considere que o alfabeto de uma sequência qualquer seja $\mathcal{X} = \{a, b, c, d, e, f\}$ e que deseja-se codificar os símbolos dessa sequência usando um alfabeto $\mathcal{A} = \{0,1\}$. Então, o código para $\mathcal{X}$, definido sobre $\mathcal{A}$, pode ser: $\mathcal{C}(a) = 000$, $\mathcal{C}(b) = 010$, $\mathcal{C}(c) = 001$, $\mathcal{C}(d) = 011$, $\mathcal{C}(e) = 111$ e $\mathcal{C}(f) = 1111$. Nesse exemplo, $L_{\mathcal{C}}(a) = 3$, $L_{\mathcal{C}}(b) = 3$, $L_{\mathcal{C}}(c) = 3$, $L_{\mathcal{C}}(d) = 3$, $L_{\mathcal{C}}(e) = 3$ e $L_{\mathcal{C}}(f) = 4$. Se for considerado que $p(a) = 0{,}15$, $p(b) = 0{,}25$, $p(c) = 0{,}1$, $p(d) = 0{,}2$, $p(e) = 0{,}1$ e $p(f) = 0{,}2$, o tamanho do código para $\mathcal{X}$, definido sobre $\mathcal{A}$ e calculado pela Equação 4.3, é: $L(\mathcal{C}) = 3 \times 0{,}15 + 3 \times 0{,}25 + 3 \times 0{,}1 + 3 \times 0{,}2 + 3 \times 0{,}1 + 4 \times 0{,}2 = 3{,}2$. (COVER; THOMAS, 2006).

Dado que cada símbolo do alfabeto $\mathcal{X}$ é mapeado para uma representação única (código), então uma função de decodificação é capaz de recuperar qualquer sequência original a partir do código $\mathcal{C}$. Porém, a função de decodificação precisará conhecer toda a

sequência binária para determinar qual símbolo da sequência original ela representa. Uma forma de evitar esse problema e tornar a decodificação instantânea é o uso de códigos prefixos.

4.1.1.1 Códigos prefixos

O princípio MDL utiliza apenas códigos prefixos (GRÜNWALD, 2005). Nesse tipo de código, nenhuma palavra-código é prefixo de outra. Portanto, ele pode ser decodificado de maneira instantânea, pois o fim do código é imediatamente reconhecido. Por exemplo, o código $\mathcal{C}_1$, dado por $\mathcal{C}_1(a) = 0$, $\mathcal{C}_1(b) = 10$, $\mathcal{C}_1(c) = 110$, $\mathcal{C}_1(d) = 1110$ e $\mathcal{C}_1(e) = 1111$, é prefixo para $\mathcal{X} = \{a, b, c, d, e\}$ com alfabeto $\mathcal{A} = \{0,1\}$. Já, o código $\mathcal{C}_2$, dado por $\mathcal{C}_2(a) = 111$, $\mathcal{C}_2(b) = 011$, $\mathcal{C}_2(c) = 010$, $\mathcal{C}_2(d) = 001$ e $\mathcal{C}_2(e) = 1111$ não é um código prefixo, pois 111 é prefixo de 1111. Um código prefixo pode ser representado por meio de árvores, onde cada nó folha é um código e o percurso entre a raíz e um dos nós folha identifica os símbolos que fazem parte do código. Um exemplo é ilustrado pela Figura 3.

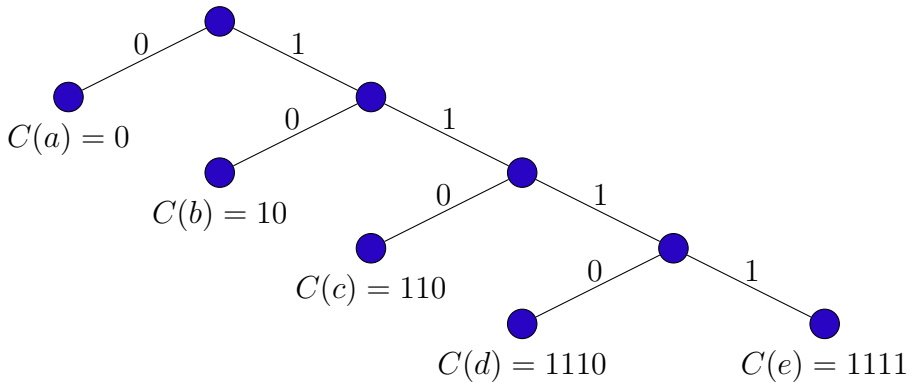


Figura 3 – Árvore binária associada ao código prefixo definido por $C(a) = 0$, $C(b) = 10$, $C(c) = 110$, $C(d) = 1110$ e $C(e) = 1111$.

Para um código prefixo $\mathcal{C}$ que utilize um alfabeto binário, como no exemplo acima, os tamanhos dos códigos ou tamanhos de descrição $L_{\mathcal{C}}(x_1), L_{\mathcal{C}}(x_2), ..., L_{\mathcal{C}}(x_n)$ devem satisfazer a desigualdade de Kraft, definida por:

$$\sum_{i=1}^{|\mathcal{X}|} 2^{-L_{\mathcal{C}}(x_i)} \leq 1. \tag{4.4}$$

Portanto, dado um código qualquer, se a soma dos tamanhos dos códigos que o compõe satisfaz a desigualdade de Kraft, pode-se garantir que esse código é um código prefixo. Segundo Cover e Thomas (2006), para demonstrar a inequação acima, considere que L_{max} é o tamanho do código mais longo. Agora, considere todos os nós folhas da árvore que estão no nível L_{max}, como no exemplo ilustrado pela Figura 3. Alguns dos nós folhas são códigos, outros são descendentes de códigos e outros não são nem códigos e nem

descendentes de códigos. Um código no nível $L_{\mathcal{C}}(x_i)$ terá $2^{L_{max}-L_{\mathcal{C}}(x_i)}$ descendentes no nível L_{max}. Esses descendentes, para serem códigos prefixos, precisam ser disjuntos. Além disso, como o número total de códigos de comprimento L_{max} é $2^{L_{max}}$, tem-se:

$$\sum_{i=1}^{|\mathcal{X}|} 2^{L_{max}-L_{\mathcal{C}}(x_i)} \leq 2^{L_{max}}, \tag{4.5}$$

que corresponde à Equação 4.4.

Dado que $\mathcal{C}$ é um código prefixo em $\mathcal{X}$ com tamanho $L_{\mathcal{C}}$, então:

$$\forall x \in \mathcal{X} : p(x) := 2^{-L_{\mathcal{C}}(x)}. \tag{4.6}$$

Como $p(x)$ é sempre positivo e menor ou igual a um, ele pode ser interpretado como uma função de probabilidade que define a distribuição correspondente a $\mathcal{C}$. Por outro lado, se uma distribuição p obedece à inequação de Kraft, existe um código prefixo que satisfaz:

$$\forall x \in \mathcal{X} : L_{\mathcal{C}}(x) := \lceil -\log p(x) \rceil, \tag{4.7}$$

onde $\lceil . \rceil$ é a operação de arredondamento de um número real para um inteiro maior mais próximo. Esse arredondamento é necessário, pois o tamanho do código, por definição, precisa ser um inteiro. Também é importante observar que $-\log p(x)$ corresponde ao código de Shannon-Fano.

Outro ponto importante é que, para uma determinada sequência Z, qualquer código que satisfaz a desigualdade de Kraft possui um comprimento médio maior ou igual à entropia dessa sequência:

$$L(\mathcal{C}) \geq H(Z). \tag{4.8}$$

A igualdade só ocorre na equação acima se $L_{\mathcal{C}}(x_i) = -\log p(x_i)$ e, nesse caso, o código é dito ser ideal. Porém, como $L_{\mathcal{C}}(x_i)$ deve ser um número inteiro e, normalmente, $p(x_i)$ possui um valor real, a igualdade geralmente não ocorre. É importante notar que quanto maior a probabilidade do símbolo x_i, menor será o tamanho do código que o representa.

A associação entre compressão e probabilidade por meio da desigualdade de Kraft é muito importante para o princípio MDL, pois a escolha do modelo com o menor tamanho de descrição pode ser feita analisando-se as distribuições de probabilidade de cada um deles.

4.2 O princípio MDL

O princípio da descrição mais simples (MDL) possui a prerrogativa de que em um problema de seleção de modelos que tenha dois ou mais candidatos, deve-se escolher aquele que possui o menor tamanho de descrição (RISSANEN, 1978; RISSANEN,

1983). Isso significa que modelos menos complexos são preferíveis (BARRON *et al.*, 1998; GRÜNWALD *et al.*, 2005). A simplicidade no princípio MDL é interpretada como o tamanho de descrição do modelo quando ele é usado para representar o dado.

O princípio MDL é similar ao princípio da mínima mensagem (MML - *minimum message length*) (WALLACE; BOULTON, 1968; WALLACE; FREEMAN, 1987), pois as duas abordagens são baseadas na ideia de que quanto mais um dado pode ser comprimido, mais conhecimento se tem em relação ao mesmo. Porém, elas possuem diferenças importantes que são apresentadas em Grünwald *et al.* (1998), Baxter e Oliver (1994) e Hansen e Yu (2001). Por exemplo, segundo Hansen e Yu (2001), o princípio MML tem focado na estimação de parâmetros, enquanto o princípio MDL abrange a modelagem estatística em geral.

O conceito por trás do princípio MDL, de escolher modelos menos complexos, é uma formalização do princípio da navalha de Occam. Essa ideia também está associada à complexidade de Kolmogorov que, em termos gerais, é o tamanho do menor programa que descreve um objeto representado por uma sequência binária (KOLMOGOROV, 1965; SOLOMONOFF, 1964; CHAITIN, 1969). Uma sequência binária que possui bastante regularidade tem uma complexidade de Kolmogorov menor do que uma sequência com menos regularidade, pois pode ser descrita por um programa menor. O menor programa que consegue descrever uma sequência binária é o que mais detectou regularidades nela, o que no contexto de aprendizado pode ser entendido como o programa que possui o maior poder preditivo sobre ela.

A Figura 4 apresenta duas sequências binárias. A primeira sequência, claramente possui mais regularidade que a segunda. Ela é formada por k repetições de 0001, enquanto a segunda é aleatória e, portanto, não possui regularidade. Diante disso, é possível criar um programa que imprime a sequência A de forma compacta, com um tamanho de descrição menor do que a sequência original, como no exemplo ilustrado na Figura 5. Por outro lado, como a sequência B não possui regularidade, qualquer programa usado para descrevê-la terá aproximadamente o mesmo tamanho de descrição dela, pois terá que imprimi-la usando uma representação literal.

A. 00010001000100010001...0001000100010001000100010001
B. 01110100110100100110...1010111010111011000101100010

Figura 4 – Sequências binárias (Fonte: Grünwald (2007, pág. 4)).

É importante mencionar que não é possível determinar o tamanho de todos os possíveis programas que podem descrever todos os conjuntos de sequências binárias e, portanto, a complexidade de Kolmogorov não é computável (LI; VITÁNYI, 2008).

Dado que o MDL incorpora o princípio da navalha de Occam, a complexidade de Kolmogorov e, ao mesmo tempo, procura selecionar o modelo que melhor representa os dados, ele gera um balanceamento entre a complexidade do modelo e a qualidade do

```
for i in range(k): print('0001',end ='')
```

(a) Programa em Python que imprime a sequência binária A da Figura 4.

```
print('01110100110100100110...10101110101110110001011000010')
```

(b) Programa em Python que imprime a sequência binária B da Figura 4.

Figura 5 – Programas em Python que imprimem as sequências binárias da Figura 4.

seu ajuste aos dados de observação. Tais características são bastante desejadas na área de aprendizado de máquina, pois naturalmente evitam o problema de sobreajustamento, já que uma hipótese altamente complexa que se ajusta extremamente bem aos dados observados, normalmente comete muitos erros em dados não conhecidos.

Matematicamente, dado um conjunto de potenciais modelos $M_1, M_2, ..., M_k$, o MDL seleciona aquele, tal que:

$$M_{mdl} = \arg\min_{M}[L(M) + L(Y|M)], \tag{4.9}$$

sendo que $L(M)$ corresponde ao tamanho da descrição do modelo M, representando sua complexidade, e $L(Y|M)$ ao tamanho da descrição do dado Y quando codificado com M, representando quão bem M descreve Y. Rissanen (1978) mostrou que $L(Y|M)$ pode ser calculado através do código de Shannon-Fano (SHANNON, 1948) e, portanto, $L(Y|M) = -\log p(Y|M)$, onde $p(Y|M)$ é a probabilidade condicional de Y dado M.

O MDL definido na Equação 4.9 foi proposto por Rissanen (1978) e Rissanen (1983) e é conhecido como *crude* MDL. De acordo com Grünwald (2005), um problema do *crude* MDL é encontrar uma boa codificação para os modelos M, pois existe um risco de que $L(M)$ se torne arbitrário, uma vez que ele pode ser muito grande para um determinado código e muito pequeno para outro.

Rissanen (1996) contornou o problema do *crude* MDL propondo uma nova versão do MDL que usa códigos universais para medir o tamanho da descrição de cada modelo. Basicamente, nessa nova versão, é empregada apenas uma parte do código com tamanho $\bar{L}(Y|M)$, ou seja, o modelo e os dados são codificados conjuntamente.

4.3 Trabalhos correlatos

Existem na literatura vários trabalhos referentes ao emprego do princípio MDL, principalmente na área de Teoria da Informação. As principais referências para o estudo das bases teóricas podem ser encontradas nos livros de Grünwald *et al.* (2005) e Grünwald (2007), além dos trabalhos de Barron *et al.* (1998) e de Hansen e Yu (2001).

Os trabalhos existentes interpretam e aplicam o princípio MDL de maneiras distintas, pois não existe um procedimento padrão para calcular o tamanho de descrição

de um modelo. Na literatura, os autores usam o princípio MDL como uma espécie de guarda-chuva para todos os métodos de inferência indutiva baseada na compressão de dados, mesmo que não possa ser diretamente interpretada em termos de minimização do tamanho de descrição. Para outros autores, o princípio MDL é qualquer método baseado na ideia de encontrar o modelo com menor tamanho de descrição que represente os dados. Muitos autores também usam o MDL para descrever um critério de seleção de modelos que é equivalente ao *Bayesian information criterion* (BIC), ou até mesmo usando uma interpretação Bayesiana, como por exemplo em Duda *et al.* (2000) e MacKay (2005). Há ainda trabalhos que associam o MDL ao método *maximum a posteriori* (MAP), como por exemplo Russell e Norvig (2010) e Mitchell (1997).

Apesar de existirem diversos métodos de aprendizado de máquina que usam o princípio MDL como mecanismo auxiliar ou estratégia secundária, não há conhecimento da existência de nenhum método de classificação, especialmente voltado para problemas multiclasse, que use o princípio MDL como o recurso principal do seu modelo de predição.

Um dos primeiros trabalhos a usar o princípio MDL em aplicações práticas de aprendizado de máquina foi o de Quinlan e Rivest (1989). Neste trabalho, ele foi usado na geração de árvores de decisão. Kohonenko (1998) também utilizou o princípio MDL na construção de árvores de decisão, porém, diferentemente do trabalho anterior, que usou o princípio MDL para auxiliar a construção dos nós de decisão da árvore, ele propôs usá-lo para auxiliar o processo de poda. Mehta *et al.* (1996) também propuseram um algoritmo de poda baseado no princípio MDL que foi usado em um método de classificação baseado em árvores de decisão chamado de SLIQ.

O princípio MDL também foi utilizado no aprendizado de redes Bayesianas. Por exemplo, Lam e Bacchus (1994) propuseram um método para aprendizado de redes Bayesianas que usa o princípio MDL e apontaram a relação deste com o método de entropia cruzada mínima. Segundo eles, isso permitiu a criação de um algoritmo heurístico de busca para redes que minimiza a entropia cruzada. Essas redes diminuem o tamanho da codificação dos dados e consequentemente, a complexidade da rede, o que possibilita a obtenção de modelos melhores na perspectiva do princípio MDL. De acordo com os autores, o algoritmo proposto gera um balanceamento entre a complexidade das redes e a acurácia obtida pelas mesmas.

Seleção de atributos é outro problema clássico de aprendizado de máquina que também já foi beneficiado por métodos baseados no princípio MDL. Em Bosin *et al.* (2006), o princípio MDL é utilizado como critério para a seleção de atributos relevantes de bases de dados de microarranjos (*micro-array*) que possuem um grande espaço de atributos e poucos exemplos. Nesse mesmo contexto, Chaitankar *et al.* (2009) propuseram um algoritmo de inferência que incorpora informação mútua, informação mútua condicional e o princípio MDL para detectar redes de regulação gênica em dados de microarranjos de DNA. A informação mútua e a informação mútua condicional são usadas para determinar

as relações regulatórias entre os genes, enquanto o princípio MDL é usado para determinar o melhor limiar de informação mútua sem a necessidade de um parâmetro de ajuste fino especificado pelo usuário. Um problema semelhante foi abordado por Tabus e Astola (2001). Eles propuseram um algoritmo para o problema de predição gênica que usa o princípio MDL para selecionar o tamanho adequado da janela de predição e também para comparar preditores com diferentes complexidades. O problema abordado pelos autores foi o processamento de microarranjos cDNA valorizado ternário para encontrar grupos de genes que provavelmente podem determinar a atividade de um gene alvo.

Problemas relacionados à morfologia também já foram tratados por abordagens que usam o princípio MDL. Por exemplo, Creutz e Lagus (2002) propuseram um método baseado no princípio MDL e outro baseado na otimização de máxima verossimilhança aplicados para a segmentação não supervisionada de palavras em morfemas. Segundo os autores, os métodos propostos obtiveram bons resultados, se comparados aos métodos estado-da-arte. Em outro trabalho, Goldsmith (2006) propôs um algoritmo que usa o princípio MDL e é aplicado ao aprendizado não supervisionado de morfologia com ênfase em linguagens tipologicamente similares às linguagens europeias e que não possuem um número médio muito alto de morfemas por palavra.

Outra área de estudo que foi beneficiada com o uso do princípio MDL foi em análise de imagens, com o trabalho de Potapov (2012). Ele propôs uma extensão do princípio MDL chamada de MDL representacional (RMDL – *representational* MDL) que torna possível a construção de métodos de análise de imagem com alta capacidade de aprendizado através da otimização automática de representações.

O princípio MDL também já foi aplicado em problemas que envolvem a classificação de séries temporais. Em Begum *et al.* (2013), foi proposto um critério livre de parâmetros baseado no princípio MDL para classificação semi-supervisionada de séries temporais que foi aplicado em várias bases de dados médicas, incluindo bases de dados de eletrocardiogramas.

Em Ince e Klawonn (2013), foi proposta uma abordagem para escolher automaticamente o nível de granularidade de atributos categóricos para um método de classificação *naïve* Bayes (NB). Nesta abordagem, o classificador NB é apresentado para o princípio MDL como um modelo. Caso o classificador NB classifique corretamente todas as instâncias, no cálculo do tamanho da descrição que será usado pelo princípio MDL, é necessário conhecer apenas o classificador NB e os valores dos atributos de cada registro. Se o NB classificar corretamente a maioria das instâncias, então é suficiente conhecer o classificador NB e as classes daqueles registros que ele classificou incorretamente. Então, o princípio MDL escolhe o modelo que tiver o menor tamanho de descrição.

Outra aplicação prática do princípio MDL foi em problemas que envolvem símbolos e glifos (letras, caracteres ou símbolos individuais) e pode ser conferido nos trabalhos de Tataw *et al.* (2013) e Li *et al.* (2014). No primeiro, os autores propuseram um

método de clusterização de glifos baseado no princípio MDL que é capaz de ignorar alguns dos dados disponíveis, tais como palavras raras ou caracteres degradados. No segundo, foi proposto um *framework* iterativo para o aprendizado não supervisionado de símbolos gráficos manuscritos, que possui como limitação o fato de poder ser aplicado apenas em bases de dados de símbolos matemáticos.

Robinet *et al.* (2011) propuseram um modelo computacional de como os humanos indutivamente identificam e agregam conceitos a partir de estímulos de baixo nível aos quais são expostos. Eles implementaram um mecanismo de *chunking* dinâmico e hierárquico em que a decisão sobre se um novo *chunk* deve ser criado é feita a partir de um critério baseado no princípio MDL. O modelo proposto foi chamado de *MDLChunker* e foi testado em um experimento onde os participantes foram apresentados a símbolos sem sentido e foram encorajados a criar conceitos de auto nível e agrupá-los. Segundo os autores, o modelo proposto conseguiu fazer predições quantitativas precisas sobre os tipos de *chunks* criados pelos participantes e o momento em que essas criações ocorreram.

Kim e Kweon (2006) propuseram um novo critério para a classificação e seleção de palavras visuais (determinar a palavra relacionada a uma imagem), que usa uma extensão do princípio MDL. Segundo os autores, o critério proposto é capaz de fornecer palavras visuais ótimas com uma acurácia suficiente na classificação.

Por fim, Almeida *et al.* (2010) e Almeida e Yamakami (2012) propuseram um método de classificação binária para a filtragem de spam em *emails* formado pela combinação do princípio MDL com fatores de confidência, que foi chamado de MDL-CF. Segundo os autores, o método proposto obteve melhores resultados que os demais métodos da literatura. Esse método foi usado nesta tese como inspiração para o desenvolvimento de um método de categorização de textos genérico, baseado no princípio MDL, que possa ser usado em problemas binários e multiclasse e em cenários de aprendizado *online* e *offline*.

4.4 Considerações finais

Neste capítulo, foram apresentados os conceitos básicos sobre Teoria da Informação com o objetivo de relacionar como o princípio MDL está associado à codificação e compressão de dados. Foi mostrado que pela inequação de Kraft é possível associar o tamanho de descrição de códigos prefixos com sua distribuição de probabilidade. Portanto, a ideia do princípio MDL de selecionar o modelo com menor tamanho de descrição pode ser generalizada para problemas que não estão associados diretamente à compressão de dados, como por exemplo problemas de classificação que também envolvem seleção de modelos.

Adicionalmente, foram apresentados os fundamentos matemáticos do princípio MDL e alguns trabalhos de aprendizado de máquina que o aplicam. Porém, não existe um modo padrão de calcular o tamanho de descrição de um modelo e, portanto, os autores têm

aplicado o princípio MDL usando diferentes interpretações. Além disso, não há nenhum método de classificação para problemas multiclasse que use o princípio MDL como recurso principal de seu modelo de predição.

5 O método MDLText

Neste capítulo, é apresentado um novo método de categorização de textos baseado no princípio MDL. O método proposto, nomeado MDLText, foi inspirado no método de classificação binária inicialmente proposto por Almeida *et al.* (2010) e Almeida e Yamakami (2012) para o problema de detecção de spam em *emails*. O método proposto tem a vantagem de poder ser utilizado em problemas de classificação de texto multiclasse e ser combinado com qualquer técnica capaz de determinar a relevância dos termos, o que o torna mais flexível para ser aplicado com sucesso em diferentes cenários. Ainda, ele possui uma função que calcula a similaridade entre o documento que está sendo classificado e as possíveis classes do problema, o que aumenta seu poder de predição.

Primeiramente, é apresentada a base matemática do método, onde é descrito como a ideia de selecionar o menor tamanho de descrição foi interpretada e usada no modelo de predição. Em seguida, são apresentados os algoritmos de treinamento e classificação e o estudo da complexidade computacional. Por fim, são apresentadas diferentes técnicas que podem ser usadas para determinar a pontuação de relevância para os termos, que é usada no cálculo do tamanho de descrição.

5.1 Base matemática

Dado um documento de texto d não rotulado, o MDLText utiliza a equação principal do princípio MDL (Equação 4.9) para predizer a classe à qual o documento pertence. O conjunto de classes $\mathcal{C} = \{c_1, c_2, \ldots, c_{|\mathcal{C}|}\}$ representa o conjunto de potenciais modelos M, enquanto d representa o dado Y. Além disso, d recebe o rótulo j correspondente à classe c_j que possui o menor tamanho de descrição em relação à d:

$$c(d) = \arg\min_{\forall c} L(d|c_j). \tag{5.1}$$

Nesta proposta, foi ignorado o tamanho de descrição das potenciais classes (modelos), pois como mencionado na Seção 4.2, ele pode se tornar arbitrário e difícil de ser calculado (RISSANEN, 1996). Portanto, foi empregada uma aplicação direta do princípio MDL que leva em conta apenas o tamanho de descrição do documento d para cada potencial classe.

O tamanho da descrição de d em relação à classe c_j é calculado multiplicando-se (1) uma penalidade $\hat{S}(d, c_j)$ para a classe c_j, (2) a soma do tamanho da descrição de todos os termos de d e (3) uma penalidade $K(t_i)$ para cada termo, conforme a equação a seguir:

$$L(d|c_j) = \hat{S}(d, c_j) \times \left[\sum_{i=1}^{|d|} L(t_i|c_j) \times K(t_i) \right]. \tag{5.2}$$

onde $|d|$ corresponde à quantidade de termos no documento d e $L(t_i|c_j)$ é o tamanho de descrição do termo t_i dada a classe c_j. Para calcular $L(t_i|c_j)$, é usada a seguinte equação, inspirada no código Shannon-Fano:

$$L(t_i|c_j) = \lceil -\log_2 \beta(t_i|c_j) \rceil. \tag{5.3}$$

O fator $\beta(t_i|c_j)$ é calculado com base no peso TF-IDF normalizado do termo t_i de cada documento do conjunto de treinamento, usando a seguinte equação:

$$\beta(t_i|c_j) = \frac{n_{c_j,t_i} + \frac{1}{|\Omega|}}{\hat{n}_{c_j} + 1}, \tag{5.4}$$

onde $n_{c_j,t_i} = \sum_{\forall d} \hat{w}(t_i, d|c_j)$ (soma dos pesos TF-IDF normalizados do termo t_i nos documentos de treinamento da classe c_j) e $\hat{n}_{c_j}$ é a soma de n_{c_j,t_i} para todos os termos que aparecem em documentos pertencentes à classe c_j. O parâmetro $|\Omega|$ é usado para preservar uma porção do tamanho de descrição para termos que nunca apareceram no conjunto de treinamento $\mathcal{D}^\star$ e que possuem, portanto, $n_{c_j,t_i} = 0$.

A Equação 5.4 é calculada da mesma maneira que um modelo de probabilidades é construído por meio da estimativa de máxima verossimilhança. Porém, em vez de usar a frequência dos termos, foi usado o peso TF-IDF. Segundo Rennie *et al.* (2003), o desempenho dos métodos que utilizam a estimativa de máxima verossimilhança pode ser melhorado quando o peso TF-IDF é usado no lugar do peso TF dos termos. A Equação 5.4 também é similar à estimativa feita pelo método NB sem considerar as probabilidades a *priori* das classes e calculando a soma dos logaritmos das probabilidades em vez de calcular o produto delas.

Como é mostrado na Figura 6, quando o valor de $|\Omega|$ é alto, o peso condicional do termo é baixo e, portanto, o tamanho da descrição do termo, quando $n_{c_j,t_i} = 0$, é alto. Então, pode-se afirmar que o parâmetro $|\Omega|$ regula como os termos com $n_{c_j,t_i} = 0$ irão contribuir no tamanho de descrição de um documento em relação a uma classe c_j. A Figura 6 também mostra que se $\hat{n}_{c_j}$ é alto, o tamanho de descrição do termo em relação a uma classe c_j também será alto quando $n_{c_j,t_i} = 0$.

Dado que $L(d|c_j)$ é claramente sensível ao valor de $|\Omega|$, é recomendável usar uma busca em grade (*grid search*) para encontrar o melhor valor para este parâmetro, que pode variar de acordo com a aplicação.

A Figura 7 ilustra como $L(t_i|c_j)$ varia de acordo com $\beta(t_i|c_j)$. Se o valor de $\beta(t_i|c_j)$ é alto, então $L(t_i|c_j)$ é baixo. É importante notar que na prática $\beta(t_i|c_j)$, em geral, tem um valor mais próximo de zero do que de um (região cinza da Figura 7a), pois problemas de categorização de texto normalmente possuem uma grande quantidade de amostras com alta dimensionalidade e representadas por vetores altamente esparsos (considerando que os documentos são representados usando o modelo espaço-vetorial).

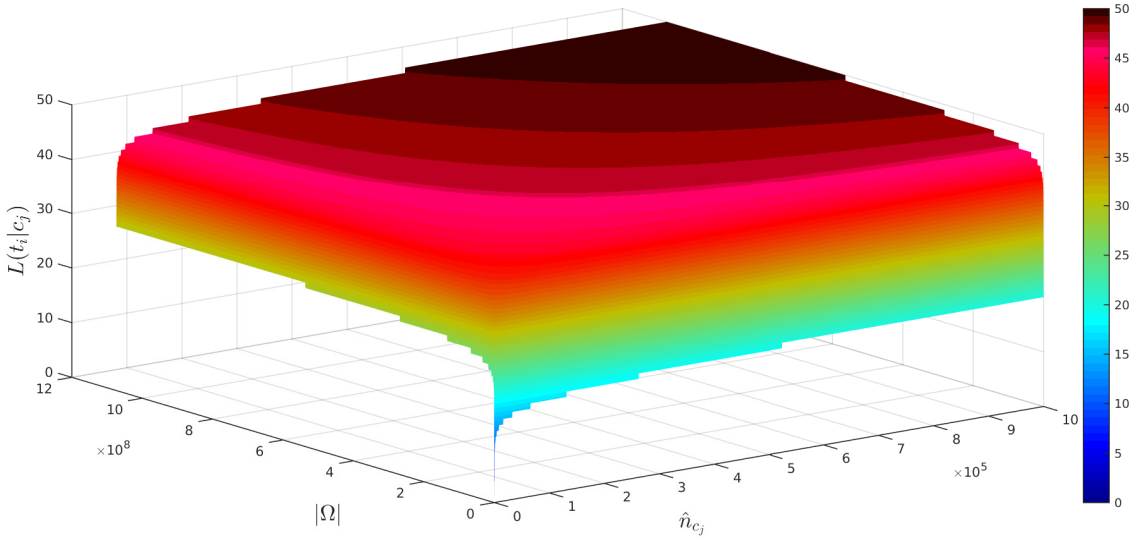
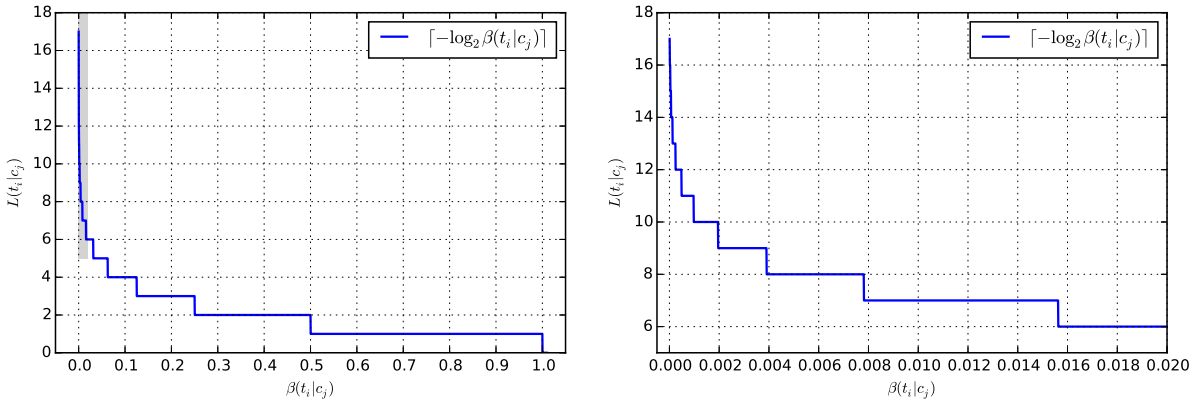


Figura 6 – Impacto de $|\Omega|$ e $\hat{n}_{c_j}$ no valor de $L(t_i|c_j)$ para os termos com $n_{c_j,t_i} = 0$.



(a) (b)

Figura 7 – Impacto de $\beta(t_i|c_j)$ no valor de $L(t_i|c_j)$. A Figura 7b mostra a região cinza da Figura 7a.

Na Equação 5.2, a função de penalidade $\hat{S}(d, c_j)$ é baseada na similaridade de cosseno (*cosine similarity*) entre o documento e um vetor protótipo da classe:

$$\hat{S}(d, c_j) = -\log_2 \left(\frac{1}{2} \times S(d, \overline{c}_j) \right). \tag{5.5}$$

A similaridade de cosseno $S(d, \overline{c}_j)$ mede o quão similar um documento d é de um vetor protótipo $\overline{c}_j$, de uma maneira análoga ao que é feito no método Rocchio (ROCCHIO, 1971; MANNING *et al.*, 2009). A similaridade $S(d, \overline{c}_j)$ varia entre zero e um e valores mais próximos a um indicam que existe grande similaridade. A função $\hat{S}(d, c_j) \geq 1$ penaliza a classe com menor similaridade em relação ao documento. Quanto menor a similaridade, maior é o valor obtido em $\hat{S}(d, c_j)$ (veja a Figura 8) e, portanto, maior é o tamanho de descrição do documento d dada a classe c_j.

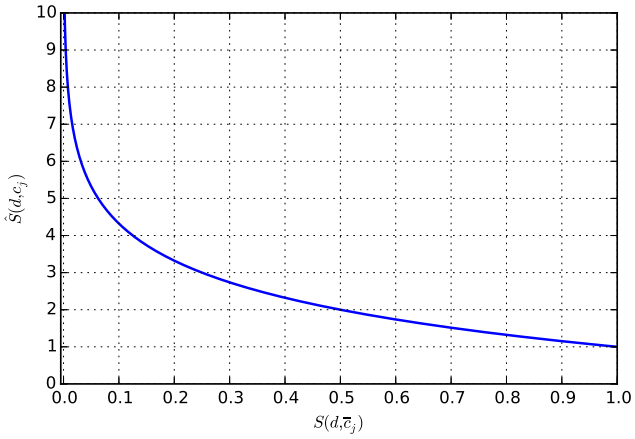


Figura 8 – Variação de $\hat{S}(d, c_j)$ de acordo com $S(d, \overline{c}_j)$.

A similaridade de cosseno pode ser calculada por:

$$S(d, \overline{c}_j) = \frac{\sum_{i=1}^{|d|} \hat{w}(t_i, d|c_j) \times \overline{c}_j(t_i)}{\|\hat{w}(:, d)\|_2 \times \|\overline{c}_j\|_2} \tag{5.6}$$

e o vetor protótipo $\overline{c}_j$ é formado pela média dos pesos dos termos. Portanto, para cada termo t_i, $\overline{c}_j(t_i)$ é:

$$\overline{c}_j(t_i) = \frac{n_{c_j,t_i}}{|\hat{\mathcal{D}}_{c_j}|} \tag{5.7}$$

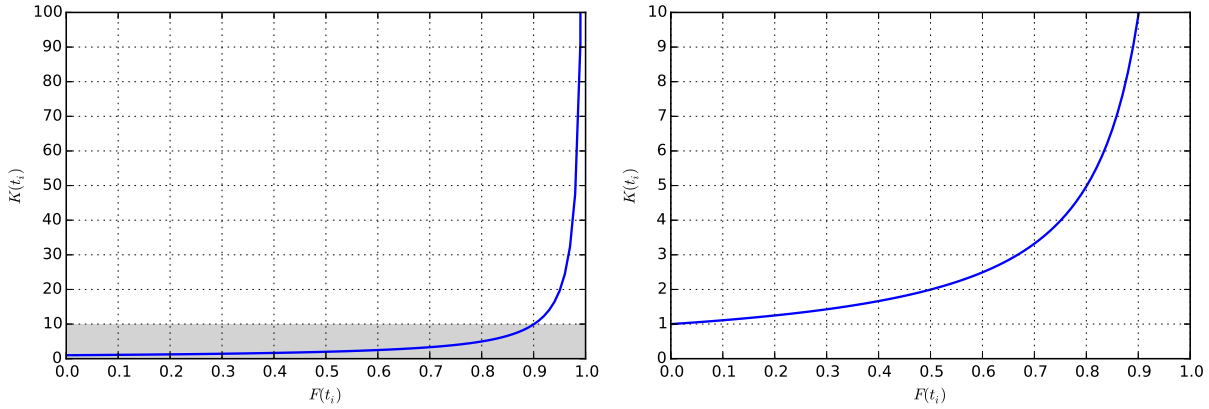
onde $|\hat{\mathcal{D}}_{c_j}|$ é a quantidade de documentos de treinamento da classe c_j.

Na Equação 5.2, a função de penalidade $K(t_i)$ associada a cada termo pode ser calculada da seguinte maneira:

$$K(t_i) = \frac{1}{(1 + \mu) - F(t_i)}, \tag{5.8}$$

onde $0 \leq F(t_i) \leq 1$ corresponde à pontuação de contribuição do termo t_i e $\mu > 0$ é uma constante usada para evitar que o denominador possa tornar-se igual a zero. Neste trabalho, empiricamente determinou-se que $\mu = 10^{-3}$. A Figura 9 mostra como a função de penalidade $K(t_i)$ varia de acordo com $F(t_i)$.

O principal objetivo ao usar a função de penalidade $K(t_i)$ é aumentar a habilidade do classificador em detectar as classes do problema. A ideia por trás dessa função é fazer com que os termos com alta relevância contribuam mais com $L(d|c_j)$. Especificamente, suponha que um termo $t^\star$ com alta pontuação de contribuição apareça em vários documentos de uma determinada classe (por exemplo, $j = 1$), porém em poucos documentos das outras classes ($j > 1$). Então, $L(t^\star|c_1)$ será menor que qualquer $L(t^\star|c_j)$ para $j > 1$. $F(t^\star)$ é alto, portanto $K(t^\star)$ também é alto. Consequentemente, $L(t^\star|c_1) \times K(t^\star)$ será bem menor que $L(t^\star|c_j) \times K(t^\star)$ para $j > 1$. Para termos com baixa representatividade ($K(t_i)$ pequeno), sua contribuição no valor de $L(d|c_j)$ será reduzida. É importante notar que um termo com alta pontuação de contribuição não possui necessariamente uma alta frequência.



(a) (b)

Figura 9 – Variação em $K(t_i)$ de acordo com $F(t_i)$. A Figura 9b mostra a região cinza da Figura 9a.

Qualquer técnica que retorne valores entre zero e um, tal como as técnicas mostradas na Seção 5.1.2, pode ser usada para calcular $F(t_i)$, desde que leve em consideração a frequência ϕ_{c_j,t_i} do termo t_i para cada classe c_j obtida do conjunto de treinamento.

Os Algoritmos 5.1 e 5.2, respectivamente, sumarizam os estágios de treinamento e teste do `MDLText`. Em ambos, considera-se que os documentos de entrada são representados por modelos espaço-vetoriais usando pesos TF-IDF.

O estágio de treinamento do `MDLText` (Algoritmo 5.1) consiste apenas na coleta de informações dos documentos de treinamento, referentes aos pesos TF-IDF dos termos e da frequência de vezes em que eles aparecem em cada classe. Tais informações são guardadas nas variáveis $\mathcal{T}$, n, $\hat{n}$, ϕ e $|\hat{\mathcal{D}}|$, que formam o modelo de predição. A variável $\mathcal{T}$ armazena a lista dos termos que aparecem nos documentos de treinamento; n armazena os valores de n_{c_j,t_i} para todas as classes em $\mathcal{C}$ e todos os termos em $\mathcal{T}$; $\hat{n}$ armazena os valores de $\hat{n}_{c_j}$ para todas as classes em $\mathcal{C}$; ϕ armazena a frequência ϕ_{c_j,t_i} para todos os termos em $\mathcal{T}$ e todas as classes em $\mathcal{C}$; e $|\hat{\mathcal{D}}|$ armazena os valores de $|\hat{\mathcal{D}}_{c_j}|$ para todas as classes em $\mathcal{C}$. Depois do treinamento, essas informações são usadas para a classificação de dados não rotulados, baseada no Algoritmo 5.2, onde é selecionada a classe com o menor tamanho de descrição.

As variáveis $\mathcal{T}$, n, $\hat{n}$, ϕ e $|\hat{\mathcal{D}}|$ compõem o modelo de predição, mas são entradas opcionais no estágio de treinamento (Algoritmo 5.1), pois o método proposto é naturalmente adaptado para o aprendizado *online* e incremental. Portanto, se as variáveis do modelo de predição são dadas como entrada no Algoritmo 5.1, elas são atualizadas com as informações obtidas pela análise dos novos documentos de treinamento.

O código fonte do método `MDLText`, assim como todas as bases de dados textuais usadas nos experimentos, estão disponíveis publicamente na plataforma GitHub no seguinte endereço *Web*: ⟨https://github.com/renatoms88/MDLText⟩.

Pseudocódigo 5.1 Estágio de treinamento do método `MDLText`

função MDL_TREINAMENTO($\mathcal{D}^\star$, $\mathcal{C}$, $\mathcal{T}$, n, $\hat{n}$, ϕ, $|\hat{\mathcal{D}}|$)
Entradas: $\mathcal{D}^\star$ (conjunto de treinamento), $\mathcal{C}$ (conjunto de todas as possíveis classes), $\mathcal{T}$ (lista de termos), n (soma dos pesos de cada termo em $\mathcal{T}$ para cada classe em $\mathcal{C}$), $\hat{n}$ (soma dos pesos em n para cada classe em $\mathcal{C}$), ϕ (frequência de cada termo em $\mathcal{T}$ para cada classe em $\mathcal{C}$) e $|\hat{\mathcal{D}}|$ (número de documentos para cada classe em $\mathcal{C}$). Os parâmetros $\mathcal{T}, n$, $\hat{n}$, ϕ e $|\hat{\mathcal{D}}|$ são opcionais.
Saídas: $\mathcal{T}$, n, $\hat{n}$, ϕ e $|\hat{\mathcal{D}}|$.

    se $\mathcal{T}$, n, $\hat{n}$, ϕ e $|\hat{\mathcal{D}}|$ não forem apresentados **então**
        $\mathcal{T} \leftarrow \emptyset$ ▷ Crie uma dicionário vazio.
        $|\hat{\mathcal{D}}_{\forall_c}| \leftarrow 0$ ▷ Inicialize o número de documentos para cada classe em $\mathcal{C}$.
    fim se
    para cada documento d em $\mathcal{D}^\star$ **faça**
        $c \leftarrow$ classe de d
        $|\hat{\mathcal{D}}_c| \leftarrow |\hat{\mathcal{D}}_c| + 1$ ▷ Número de documentos da classe c.
        para cada termo t_i in d **faça**
            se t_i não estiver em $\mathcal{T}$ **então**
                $\mathcal{T} \leftarrow \mathcal{T} \cup t_i$ ▷ Insira o termo t_i no dicionário.
                $n_{\forall_c, t_i} \leftarrow 0$ ▷ Inicialize a soma dos pesos de t_i para cada classe em $\mathcal{C}$.
                $\hat{n}_{\forall_c} \leftarrow 0$ ▷ Inicialize a soma dos pesos em n para cada classe em $\mathcal{C}$.
                $\phi_{\forall_c, t_i} \leftarrow 0$ ▷ Inicialize a frequência de t_i para cada classe em $\mathcal{C}$.
            fim se
            $n_{c,t_i} \leftarrow n_{c,t_i} + \hat{w}(t_i, d)$ ▷ Soma dos pesos de t_i na classe c (n_{c,t_i} é uma posição de n).
            $\hat{n}_c \leftarrow \hat{n}_c + n_{c,t_i}$ ▷ Soma dos pesos em dos pesos em n para a classe c ($\hat{n}_c$ é uma posição de $\hat{n}$).
            $\phi_{c,t_i} \leftarrow \phi_{c,t_i} + 1$ ▷ Frequência do termo t_i na classe c (ϕ_{c,t_i} é uma posição de ϕ).
        fim para
    fim para
    retorne $\mathcal{T}$, n, $\hat{n}$, ϕ e $|\hat{\mathcal{D}}|$
fim função

5.1.1 Análise da complexidade computacional

O estágio de treinamento (Algoritmo 5.1) requer apenas uma passagem completa para cada documento. Portanto, com $|\mathcal{D}^\star|$ documentos e $|\bar{d}|$ termos, onde $|\bar{d}|$ corresponde ao número médio de termos por documento, a complexidade é de ordem linear $\mathcal{O}(|\mathcal{D}^\star| \times |\bar{d}|)$.

No estágio de classificação (Algoritmo 5.2), para cada documento não rotulado d, o `MDLText` calcula $K(t_i)$ examinando a frequência de cada termo em d para cada classe. Em seguida, o valor de cada termo em cada classe é usado para calcular o tamanho de descrição de d. Finalmente, o valor de cada termo para cada classe é também usado para calcular $\hat{S}(d, c_j)$ e para penalizar o tamanho de descrição de d. Consequentemente,

Pseudocódigo 5.2 Estágio de classificação do método `MDLText`

1: **função** MDL_CLASSIFICAÇÃO(d, $\mathcal{C}$, $\mathcal{T}$, n, $\hat{n}$, ϕ e $|\hat{\mathcal{D}}|$)
2: **Entradas:** d (documento não rotulado), $\mathcal{C}$ (conjunto de todas as possíveis classes), $\mathcal{T}$ (dicionário dos termos obtido no estágio de treinamento), n (soma dos pesos de cada termo em $\mathcal{T}$ para cada classe em $\mathcal{C}$), $\hat{n}$ (soma dos pesos em n para cada classe em $\mathcal{C}$), ϕ (frequência de cada termo em $\mathcal{T}$ para cada classe em $\mathcal{C}$) e $|\hat{\mathcal{D}}|$ (número de documentos para cada classe em $\mathcal{C}$).
3: **Saída**: $c(d)$ (a classe predita para o documento d).

4: **para cada** termo t_i em d **faça**
5: **se** t_i estiver em $\mathcal{T}$ **então**
6: $F(t_i) \leftarrow$ pontuação_do_termo$(t_i, \phi_{\forall_c, t_i})$
7: **senão**
8: $F(t_i) \leftarrow 0$
9: **fim se**
10: Baseado em $F(t_i)$, use a Equação 5.8 para calcular $K(t_i)$.
11: **fim para**
12: **para cada** classe c_j em $\mathcal{C}$ **faça**
13: $L(d|c_j) \leftarrow 0$
14: **para cada** termo t_i em d **faça**
15: **se** t_i não estiver em $\mathcal{T}$ **então**
16: $n_{c_j,t_i} \leftarrow 0$
17: $\hat{n}_{c_j} \leftarrow 0$
18: **fim se**
19: Baseado em n_{c_j,t_i} e $\hat{n}_{c_j}$, use a Equação 5.4 para calcular $\beta(t_i|c_j)$.
20: Baseado em $\beta(t_i|c_j)$, use a Equação 5.3 para calcular $L(t_i, c_j)$.
21: $L(d|c_j) \leftarrow L(d|c_j) + L(t_i, c_j) \times K(t_i)$. ▷ Acumule a soma dos tamanhos de descrição dos termos penalizado pelas suas pontuações. Esta linha de código é uma parte da Equação 5.2.
22: Baseado em n_{c_j,t_i} e $|\hat{\mathcal{D}}_{c_j}|$, use a Equação 5.7 para calcular $\overline{c}_j(t_i)$ (essa variável é uma posição do vetor protótipo $\overline{c}_j$).
23: **fim para**
24: **fim para**
25: **para cada** classe c_j em $\mathcal{C}$ **faça**
26: Calcule a similaridade $S(d, \overline{c}_j)$ entre d e $\overline{c}_j$ usando a Equação 5.6.
27: Baseado em $S(d, \overline{c}_j)$, use a Equação 5.5 para calcular $\hat{S}(d, c_j)$.
28: $L(d|c_j) \leftarrow \hat{S}(d, c_j) \times L(d|c_j)$ ▷ Essa linha de código é uma parte da Equação 5.2 iniciada na linha 21.
29: **fim para**
30: $c(d) = \arg\min_{\forall c} L(d|c_j)$ ▷ O rótulo da classe com o menor tamanho de descrição.
31: **retorne** $c(d)$
32: **fim função**

a complexidade computacional é da ordem de $\mathcal{O}(|\mathcal{C}| \times |\bar{d}| + |\mathcal{C}| \times |\bar{d}| + |\mathcal{C}| \times |\bar{d}|)$ ou $\mathcal{O}(3 \times |\mathcal{C}| \times |\bar{d}|)$, onde $|\mathcal{C}|$ corresponde ao número de classes. Dado que $3 \times |\mathcal{C}|$ é constante, a complexidade do estágio de classificação é assintoticamente de ordem linear $\mathcal{O}(|\bar{d}|)$.

5.1.2 Técnicas para calcular a pontuação dos termos

Conforme mencionado anteriormente, o `MDLText` pode ser combinado com técnicas de atribuição de relevância de termos. Elas retornam um valor que tenta refletir a importância de cada termo na separação das classes do problema.

As técnicas avaliadas foram as seguintes: fatores de confidência (CF – *confidence factors*) (ASSIS *et al.*, 2006), seletor de atributos distintivos (DFS – *distinguishing feature selector*) (UYSAL; GUNAL, 2012), ganho de informação (IG – *information gain*) (SEBASTIANI, 2002; YANG; PEDERSEN, 1997), chi-quadrado (*chi-square*) ou χ^2 (SEBASTIANI, 2002; YANG; PEDERSEN, 1997), coeficiente NGL (Ng-Goh-Low) (NG *et al.*, 1997), coeficiente GSS (Galavotti-Sebastiani-Simi) (GALAVOTTI *et al.*, 2000) e razão de chances (OR – *odds ratio*) (SEBASTIANI, 2002; YANG; PEDERSEN, 1997). Cada uma dessas funções está sumarizada na Tabela 7.

Tabela 7 – Funções de cálculo de relevância de atributos.

Técnica	Equação
CF	$F(t_i) = \frac{1}{\lvert\mathcal{C}\rvert-1} \times \sum_{\forall j \lvert j \neq \tau} \frac{\left(\frac{\left(\phi_{c_\tau,t_i} - \phi_{c_j,t_i}\right)^2 + \left(\phi_{c_\tau,t_i} \times \phi_{c_j,t_i}\right) - \frac{\lambda_1}{\phi_{c_\tau,t_i} + \phi_{c_j,t_i}}}{\left(\phi_{c_\tau,t_i} + \phi_{c_j,t_i}\right)^2} \right)^{\lambda_2}}{1 + \left(\frac{\lambda_3}{\phi_{c_\tau,t_i} + \phi_{c_j,t_i}} \right)}$
DFS	$F(t_i) = \sum_{\forall j} \frac{p(c_j \lvert t_i)}{p(\bar{t}_i \lvert c_j) + p(t_i \lvert \bar{c}_j) + 1}$
IG	$F(t_i) = \sum_{\forall j} p(t_i, c_j) \times \log \frac{p(t_i,c_j)}{p(t_i) \times p(c_j)} + p(\bar{t}_i, c_j) \times \log \frac{p(\bar{t}_i,c_j)}{p(\bar{t}_i) \times p(c_j)}$
χ^2	$F(t_i, c_j) = \frac{\lvert\mathcal{D}^\star\rvert \times [p(t_i,c_j) \times p(\bar{t}_i,\bar{c}_j) - p(t_i,\bar{c}_j) \times p(\bar{t}_i,c_j)]^2}{p(t_i) \times p(\bar{t}_i) \times p(c_j) \times p(\bar{c}_j)}$
NGL	$F(t_i, c_j) = \frac{\sqrt{\lvert\mathcal{D}^\star\rvert} \times [p(t_i,c_j) \times p(\bar{t}_i,\bar{c}_j) - p(t_i,\bar{c}_j) \times p(\bar{t}_i,c_j)]}{\sqrt{p(t_i) \times p(\bar{t}_i) \times p(c_j) \times p(\bar{c}_j)}}$
GSS	$F(t_i, c_j) = p(t_k, c_i) \times p(\bar{t}_k, \bar{c}_i) - p(t_k, \bar{c}_i) \times p(\bar{t}_k, c_i)$
OR	$F(t_i, c_j) = \frac{p(t_i,c_j) \times [1 - p(t_i,\bar{c}_j)]}{[1 - p(t_i,c_j)] \times p(t_i,\bar{c}_j)}$

Na Tabela 7, na função da técnica CF, τ corresponde ao índice da classe mais frequente; $j = 1, \ldots, \lvert\mathcal{C}\rvert$ são os índices das $\lvert\mathcal{C}\rvert$ classes; ϕ_{c_τ,t_i} é o número de documentos que possuem o termo t_i e que pertencem à classe mais frequente e ϕ_{c_j,t_i} é a quantidade de documentos que possuem o termo t_i e que pertencem à classe c_j; e $\lambda_1, \lambda_2, \lambda_3$ são constantes que ajustam a velocidade de decaimento do fator de confidência. Os valores usados neste trabalho foram $\lambda_1 = 0{,}25$, $\lambda_2 = 10{,}0$, e $\lambda_3 = 8{,}0$, conforme proposto originalmente por Assis *et al.* (2006). É importante ressaltar que a técnica CF foi originalmente concebida apenas para problemas binários, mas neste trabalho ela foi estendida para tratar problemas multiclasse.

Nas demais funções, $p(c_j|t_i)$ corresponde à probabilidade condicional da classe c_j, dada a presença do termo t_i; $p(\bar{t}_i|c_j)$ é a probabilidade condicional da ausência de t_i, dada a classe c_j; $p(t_i|\bar{c}_j)$ corresponde à probabilidade condicional do termo t_i, dadas as classes diferentes de c_j; $p(\bar{t}_i|\bar{c}_j)$ corresponde à probabilidade condicional da ausência do termo t_i, dadas as classes diferentes de c_j; e $|\mathcal{D}^\star|$ é a quantidade de documentos de treinamento.

A seguir, são apresentadas as faixas de valores que podem ser retornadas pelas funções apresentadas na Tabela 7.

- *CF e IG*: retornam valores entre zero e um. Valores mais próximos de um indicam que o termo tem boa contribuição na identificação da classe do problema.
- *DFS*: retorna valores entre 0 e 0,5. Valores mais próximos de 0,5 significam que o termo contribui mais para a identificação da classe do que termos com valores mais próximos de 0.
- χ^2: tem valor próximo de zero quando o termo t_i aparece frequentemente em muitas classes e tem valor zero quando o termo t_i e a classe c_j são independentes. Por outro lado, quando a ocorrência do termo t_i e da classe c_j é altamente dependente, o χ^2 tem valor alto;
- *NGL e GSS*: se o termo t_i aparece com frequência em documentos da classe c_j, o valor será positivo. Se o termo aparece com frequência para as outras classes, o valor será negativo.
- *OR*: gera valores entre zero e $+\infty$. O valor um é neutro e valores próximos ao $+\infty$ indicam que o termo t_i aparece com frequência para a classe c_j, enquanto aparece com pouca frequência em documentos de outras classes. Por outro lado, valores próximos a zero indicam que o termo t_i aparece com pouca frequência em documentos da classe c_j, enquanto ocorre com frequência nas outras classes. Para que a OR não gere valor infinito, no cálculo das probabilidades, somou-se um ao número de documentos de treinamento de cada classe e à frequência de aparecimento do termo em cada classe.

As técnicas χ^2, NGL, GSS e OR não atribuem um valor global para cada termo. Elas atribuem um valor para o termo t_i relativo a cada classe. Diante disso, para essas técnicas foi calculada uma pontuação global para o termo, selecionando o maior valor, o que pode ser representado pela seguinte equação:

$$F(t_i) = \max_{i=1}^{|\mathcal{C}|} F(t_i, c_j), \tag{5.9}$$

Conforme mencionado anteriormente, o valor de $F(t_i)$ deve estar no intervalo $0 \leq F(t_i) \leq 1$. Para as técnicas que retornam um valor fora desse intervalo, foi realizada

uma normalização. Para isso foi usada a seguinte equação:

$$F(t_i) = \frac{F(t_i) - minF}{maxF - minF}, \tag{5.10}$$

onde $maxF$ e $minF$ são, respectivamente, o maior e o menor valor que poderia ser retornado por $F(t_i)$, dada a quantidade de exemplos de treinamento de cada classe.

Com o intuito de oferecer uma explicação didática das diferenças entre os diversos métodos de atribuição de pontuação para os termos, a Tabela 8 apresenta a pontuação de 10 termos em um problema de classificação de 4 classes, cada uma com 100 documentos. As primeiras quatro colunas apresentam a frequência de cada termo $(t_i, \forall i \in \{1 \dots 10\})$ nos documentos de cada classe $(c_j, \forall j \in \{1 \dots 4\})$. É importante notar que as frequências foram atribuídas empiricamente.

Tabela 8 – Pontuação dos termos calculada pelas técnicas CF, DFS, IG, χ^2, OR, NGL e GSS.

	c_1	c_2	c_3	c_4	CF	DFS	IG	χ^2	OR	NGL	GSS
t_1	100	0	0	0	0.926	1.000	0.811	1.000	1.000	1.000	1.000
t_2	100	100	0	0	0.617	0.500	1.000	0.333	0.362	0.577	0.667
t_3	100	100	100	0	0.309	0.200	0.811	1.000	0.182	0.333	0.333
t_4	100	100	100	100	0.000	0.000	0.000	0.000	0.000	0.000	0.000
t_5	50	50	50	50	0.000	0.000	0.000	0.000	0.126	0.000	0.000
t_6	10	0	0	0	0.554	0.053	0.051	0.077	0.587	0.277	0.100
t_7	1	0	0	0	0.006	0.005	0.005	0.008	0.299	0.087	0.010
t_8	100	50	20	10	0.019	0.305	0.445	0.407	0.409	0.638	0.733
t_9	100	50	20	1	0.231	0.359	0.534	0.446	0.433	0.668	0.763
t_{10}	80	50	20	10	0.010	0.179	0.243	0.222	0.361	0.471	0.533

Pode ser observado que todos os métodos atribuem uma pontuação alta para termos que aparecem em muitos documentos de uma única classe (t_1). Além disso, as técnicas CF e OR favorecem termos que aparecem em poucos documentos que pertencem a uma única classe (t_6), enquanto as técnicas DFS, IG, χ^2 e GSS retornam uma baixa pontuação para o mesmo caso. A técnica NGL gera um valor intermediário nesta situação, se comparada às outras técnicas. Em resumo, enquanto as técnicas CF e OR favorecem a exclusividade de um termo em documentos de uma dada classe, as técnicas DFS, IG, χ^2 e GSS tentam balancear a exclusividade e a frequência de um termo em documentos que pertencem a mesma classe.

Outro ponto a ser notado é que a técnica IG atribui a pontuação máxima para termos que aparecem com bastante frequência em metade das classes, enquanto aparecem com pouca frequência na outra metade (t_2). Já, os termos que aparecem com exclusividade para uma única classe, recebem uma pontuação inferior ao que é atribuído pelas outras técnicas (t_1). Além disso, a técnica IG e a χ^2 atribuem uma pontuação alta para termos que aparecem com frequência na maioria das classes, mas aparecem com

pouca frequência em apenas uma classe (t_3), ou seja, elas consideram que a ausência do termo é uma informação importante.

Os cenários representados pelos termos t_4 e t_5 também merecem destaque. A maioria das técnicas atribuíram a mesma pontuação para esses termos. Porém, a técnica OR atribuiu uma pontuação zero para o termo t_4, enquanto deu uma pontuação maior para o termo t_5. Isso mostra que a técnica OR só considera um termo irrelevante quando ele aparece em todos os documentos de treinamento e na mesma quantidade de vezes para todas as classes. Por outro lado, para as demais técnicas, se a frequência de ocorrência do termo é igual para todas as classe, ele é considerado irrelevante e recebe pontuação zero, mesmo que não tenha ocorrido em todos os documentos.

5.2 Considerações finais

Neste capítulo, foi apresentado o `MDLText`, um novo método de categorização de texto baseado no princípio MDL. O método proposto é multiclasse e pode ser usado em problemas reais e dinâmicos, já que seu aprendizado é incremental. Além disso, ele é simples de ser implementado e também é rápido, pois possui complexidade linear, tanto no estágio de treinamento, quanto na predição.

O método proposto realiza seleção de modelo com base no princípio MDL, ou seja, prioriza aquele que apresenta o menor tamanho de descrição. Os possíveis modelos avaliados pelo `MDLText` são as possíveis classes do problema. O tamanho da descrição de cada classe leva em conta a (1) similaridade entre o documento que está sendo classificado e um protótipo da classe, (2) uma penalidade para cada termo do documento baseado em quanto cada termo pode ajudar na identificação da classe e (3) um peso condicional de cada termo relativo a todos os outros termos que já apareceram em documentos da classe.

As penalidades atribuídas às classes com base na similaridade entre os protótipos delas e do documento que será classificado ($\hat{S}(d, c_j)$) torna o método mais robusto, pois o cálculo do tamanho de descrição deixa de considerar que os atributos são independentes.

A penalidade dada a cada termo ($K(t_i)$) faz com que os termos mais relevantes tenham maior contribuição no cálculo do tamanho de descrição das classes e ao mesmo tempo não menospreza as informações carregadas por termos menos relevantes. Além disso, o método proposto flexibiliza o cálculo da pontuação de cada termo que é usada para o cálculo da penalidade $K(t_i)$.

A interação entre as três partes que compõe o cálculo do tamanho de descrição das classes no `MDLText` (Equação 5.2) pode fazer o método ter maior sucesso em diferentes problemas de categorização de texto. Por exemplo, em problemas onde a similaridade entre as classes e o documento que está sendo classificado contenha pouca informação que ajude na predição da classe, os pesos condicionais dos termos e as penalidades dos termos podem ser suficientes para a identificação da classe correta. Em outros problemas, a similaridade

entre as classes e o documento-alvo pode ser suficiente para a classificação correta, mesmo que os pesos condicionais dos termos e as penalidades atribuídas aos termos não ajudem.

6 Avaliação experimental

Neste capítulo, são detalhados os experimentos realizados para avaliar o desempenho do `MDLText`. Primeiramente, são descritas as configurações adotadas e as bases de dados utilizadas. Em seguida, são apresentados os resultados nos cenários de aprendizado *offline* e *online*.

Inicialmente, foi analisado o desempenho do `MDLText` usando diferentes técnicas para calcular a pontuação dos termos. Posteriormente, a melhor técnica foi usada como parâmetro do `MDLText` e seus resultados foram comparados aos obtidos por métodos tradicionais de aprendizado *offline* e *online*, treinados por documentos de texto representados pelos três principais esquemas de pesos: TF, TF-IDF e binário (Seção 3.1.3).

6.1 Metodologia

Foram realizados experimentos com 45 bases de dados consideradas *benchmarks* na área de categorização de textos. Elas são públicas, de grande porte e abragem diversos domínios, tais como documentos médicos e científicos, análise de sentimentos, notícias, emails, páginas Web, entre outros (ROSSI *et al.*, 2016; ZHANG *et al.*, 2016). As bases de dados utilizadas e as principais estatísticas sobre elas são apresentadas na Tabela 9, onde $|\mathcal{D}|$ é a quantidade de documentos de cada base, $|\mathcal{T}|$ corresponde ao número de atributos (tamanho do vocabulário), $|S|$ indica o grau de esparsidade (porcentagem de valores iguais a zero no modelo espaço-vetorial usado para representar os documentos), ψ e IQR são, respectivamente, a mediana do número de termos por documento e a amplitude interquartil (do inglês, *interquartile range*) do número de termos por documento e $|\mathcal{C}|$ é o número de classes. A última coluna apresenta a quantidade de documentos de cada classe.

Tabela 9 – Bases de dados textuais usadas nos experimentos (**Continua**).

Base de dados	$\|\mathcal{D}\|$	$\|\mathcal{T}\|$	$\|S\|$	ψ	IQR	$\|\mathcal{C}\|$	Tamanho das classes
20Newsgroups	18.828	45.433	99,8%	56,0	51,0	20	628, 775, 799, 910, 940, 961, 972, 973, 980, 981, 982, 985, 987, 990, 990, 991, 994, 994, 997, 999
7sectors	4.581	41.718	99,7%	102,0	105,0	7	300, 355, 399, 515, 949, 964, 1.099
ACM	3.493	60.767	98,8%	712,0	261,0	40	50, 69, 70, 71, 71, 71, 71, 72, 74, 75, 80, 81, 82, 82, 83, 84, 85, 86, 87, 89, 90, 91, 91, 92, 93, 93, 95, 96, 96, 98, 98, 98, 101, 102, 103, 104, 104, 104, 105, 106
CSTR	299	1.725	96,9%	52,0	31,8	4	25, 46, 100, 128
Dmoz-Business	18.500	8.302	99,9%	11,0	7,0	37	500 (cada classe)
Dmoz-Computers	9.500	5.010	99,8%	10,0	7,0	19	500 (cada classe)
Dmoz-Health	6.500	4.216	99,7%	12,0	6,0	13	500 (cada classe)
Dmoz-Science	6.000	4.820	99,8%	11,0	7,0	12	500 (cada classe)

Tabela 9 – Bases de dados textuais usadas nos experimentos (**Continuação**).

Base de dados	$\|\mathcal{D}\|$	$\|\mathcal{T}\|$	$\|S\|$	ψ	IQR	$\|\mathcal{C}\|$	Tamanho das classes
Dmoz-Sports	13.500	5.681	99,8%	12,0	5,0	27	500 (cada classe)
Enron	13.199	18.193	99,7%	27,0	43,0	20	309, 339, 367, 370, 379, 397, 407, 420, 489, 526, 609, 657, 715, 715, 897, 1.022, 1.108, 1.122, 1.159, 1.192
Fbis	2.463	2.000	92,0%	112,0	110,8	17	38, 43, 46, 46, 46, 48, 65, 92, 94, 119, 121, 125, 139, 190, 358, 387, 506
Industry-Sector	8.817	21.489	99,6%	57,0	84,0	12	100, 282, 354, 397, 495, 505, 557, 635, 947, 959, 991, 2.595
Irish economic	1.660	8.658	98,7%	106,0	72,0	3	431, 574, 655
La1S	3.204	13.195	98,9%	104,0	155,5	6	273, 341, 354, 555, 738, 943
La2S	3.075	12.432	98,8%	112,0	157,0	6	248, 301, 375, 487, 759, 905
Latimes	6.279	10.019	99,6%	30,0	45,0	6	521, 642, 729, 1.042, 1.497, 1.848
NFS	10.524	3.887	99,8%	6,0	3,0	16	130, 201, 307, 345, 355, 402, 442, 524, 603, 647, 739, 889, 990, 1.202, 1.339, 1.409
New3s	9.558	26.832	99,1%	160,0	164,0	44	104, 105, 106, 110, 110, 115, 116, 120, 123, 124, 126, 130, 136, 139, 139, 141, 153, 159, 161, 171, 171, 174, 179, 181, 187, 196, 198, 211, 218, 238, 243, 253, 270, 276, 278, 281, 306, 326, 328, 330, 369, 493, 568, 696
Oh0	1.003	3.182	98,3%	52,0	27,0	10	51, 56, 57, 66, 71, 76, 115, 136, 181, 194
Oh10	1.050	3.238	98,3%	55,0	25,0	10	52, 60, 61, 70, 87, 116, 126, 148, 165, 165
Oh15	913	3.100	98,1%	59,0	25,0	10	53, 56, 56, 66, 69, 98, 98, 106, 154, 157
Oh5	918	3.012	98,2%	54,0	23,0	10	59, 61, 61, 72, 74, 85, 93, 120, 144, 149
Ohscal	11.162	11.465	99,5%	60,0	26,0	10	709, 764, 864, 1.001, 1.037, 1.159, 1.260, 1.297, 1.450, 1.621
Opinosis	6.457	2.692	99,7%	7,0	5,0	51	45, 45, 48, 51, 52, 56, 57, 59, 59, 60, 62, 64, 64, 67, 73, 76, 79, 80, 82, 86, 87, 88, 89, 90, 92, 96, 103, 104, 110, 111, 112, 116, 119, 125, 130, 142, 145, 153, 155, 157, 159, 166, 171, 181, 191, 241, 266, 284, 304, 377, 528
Pubmed-Cancer	65.991	28.328	99,7%	71,0	24,0	12	104, 433, 557, 926, 2.220, 2.407, 2.848, 6.676, 7.390, 9.177, 15.603, 17.650
Pubmed-Cancer-2000	18.355	14.424	99,6%	67,0	31,0	12	139, 483, 631, 1.102, 2.000, 2.000, 2.000, 2.000, 2.000, 2.000, 2.000, 2.000
RCV1	23.983	60.355	99,8%	81,0	81,0	4	2.032, 5.746, 6.262, 9.943
RCV2-Italian	13.555	18.937	99,7%	55,0	22,0	4	281, 1.581, 3.827, 7.866
RCV2-Portuguese	3.974	9.341	99,1%	74,0	45,0	4	50, 129, 842, 2.953
RCV2-Spanish	12.388	19.472	99,7%	56,0	23,0	4	30, 1.038, 1.324, 9.996
Re0	1.504	2.886	98,2%	34,0	50,5	13	11, 15, 16, 20, 37, 38, 39, 42, 60, 80, 219, 319, 608
Re1	1.657	3.758	98,6%	41,0	38,0	25	10, 13, 15, 17, 18, 18, 19, 19, 20, 20, 27, 31, 31, 32, 37, 42, 48, 50, 60, 87, 99, 106, 137, 330, 371
Re8	7.674	8.900	99,6%	25,0	28,0	8	51, 144, 271, 293, 326, 374, 2.292, 3.923
Reuters	8.246	20.529	99,7%	43,0	32,0	10	116, 143, 158, 161, 285, 307, 361, 408, 2.362, 3.945
Reviews	4.069	22.926	99,2%	175,0	140,0	5	137, 412, 999, 1.133, 1.388
Techtc300-1092-135724	1.008	30.903	99,4%	120,0	178,0	2	499, 509
Techtc300-1092-789236	1.097	31.623	99,3%	97,0	143,0	2	509, 588
Tr11	414	6.429	95,6%	145,0	176,0	9	6, 11, 20, 21, 29, 52, 69, 74, 132
Tr12	313	5.804	95,3%	149,0	167,0	8	9, 29, 29, 30, 34, 35, 54, 93
Tr21	336	7.902	94,1%	248,0	248,0	6	4, 9, 16, 35, 41, 231
Tr23	204	5.832	93,4%	169,5	188,0	6	6, 11, 15, 36, 45, 91

Tabela 9 – Bases de dados textuais usadas nos experimentos (**Conclusão**).

Base de dados	$\|\mathcal{D}\|$	$\|\mathcal{T}\|$	$\|S\|$	ψ	IQR	$\|\mathcal{C}\|$	Tamanho das classes
Tr31	927	10.128	97,3%	173,0	153,8	7	2, 21, 63, 111, 151, 227, 352
Tr41	878	7.454	97,4%	149,0	147,0	10	9, 18, 26, 33, 35, 83, 95, 162, 174, 243
Tr45	690	8.261	96,6%	199,0	189,0	10	14, 18, 36, 47, 63, 67, 75, 82, 128, 160
Trec7-3000	6.000	100.463	99,8%	57,0	88,0	2	3.000 (cada classe)

Para dar credibilidade aos resultados e tornar os experimentos reprodutíveis, abaixo são apresentadas as configurações adotadas, incluindo o pré-processamento, ajuste de parâmetros, treinamento, classificação e medidas de desempenho.

6.1.1 Pré-processamento e *tokenização*

Os documentos das bases de dados *7sectors* são páginas HTML. Por isso, foi feita a extração do conteúdo dessas páginas (*parser*) por meio da biblioteca `Beautiful Soup`[1] disponível para Python.

Para a base de dados *Reuters-21578* foram removidos os documentos que não são rotulados e também aqueles que são rotulados como pertencentes a mais de um tópico ou classe. Então, foram selecionados os documentos dos dez tópicos mais frequentes.

As bases de dados *RCV1* e *RCV2* contém documentos organizados em três categorias: *industry*, *region*, e *codes*. Apenas a categoria *codes* foi analisada. Essa categoria é organizada em quatro grupos hierárquicos: CCAT (*Corporate/Industrial*), ECAT (*Economics*), GCAT (*Government/Social*) e MCAT (*Markets*). Foram removidos os documentos que não são atribuídos a nenhum grupo hierárquico ou aqueles associados a mais de um grupo. Além disso, para a base de dados *RCV1*, foram selecionados apenas os documentos publicados entre 20 de agosto de 1996 e 04 de setembro de 1996.

Para as bases de dados *7sectors*, *TechTC-300*, *Reuters-21578*, *RCV1* e *RCV2*, todos os textos foram convertidos para letras minúsculas. Além disso, foi aplicada *estemização* e foram removidas as *stopwords*, utilizando-se a biblioteca `NLTK (Natural Language Toolkit)`[2] disponível para Python. Na etapa de *tokenização*, os delimitadores utilizados foram quaisquer caracteres não alfanuméricos.

As demais bases de dados foram descritas e preprocessadas por Rossi *et al.* (2013) e Rossi *et al.* (2016). Elas foram disponibilizadas pelos autores na forma de modelos espaço-vetoriais gerados após aplicarem *estemização* e removerem as *stopwords*.

[1] A biblioteca `Beautiful Soup` está disponível em ⟨http://www.crummy.com/software/BeautifulSoup/⟩. Acessado em: 19/01/2017.

[2] A biblioteca `NLTK (Natural Language Toolkit)` está disponível em ⟨http://www.nltk.org/⟩. Acessado em: 19/01/2017.

6.1.2 Avaliação

O `MDLText` foi testado em duas tarefas de classificação: aprendizado *offline* e aprendizado *online*. No aprendizado *offline*, todos os exemplos de treinamento são apresentados ao método de classificação de uma única vez e este cria um modelo de predição global para classificar dados não conhecidos. Por outro lado, no aprendizado *online*, é apresentado um exemplo por vez e o modelo de predição é criado de forma incremental.

O objetivo desses experimentos foi verificar o desempenho geral do `MDLText` e, portanto, os resultados do método proposto foram comparados com os resultados de vários métodos estado-da-arte em categorização de texto em ambos os cenários.

Para comparar os resultados, foi usada a macro F-medida que é amplamente utilizada na literatura em problemas de classificação multiclasse. Para maiores detalhes dessa medida de desempenho, consulte a Seção 2.4.

6.1.3 Métodos de aprendizado *offline*

Neste trabalho, os resultados obtidos pelo `MDLText` foram comparados com os resultados obtidos pelos seguintes métodos consolidados de aprendizado *offline*: NB multinomial (M.NB) (MCCALLUM; NIGAM, 1998), NB Bernoulli (B.NB) (MCCALLUM; NIGAM, 1998), Rocchio (ROCCHIO, 1971; MANNING *et al.*, 2009), SVM (BOSER *et al.*, 1992; CORTES; VAPNIK, 1995), KNN (COVER; HART, 1967), DT (CART) (BREIMAN *et al.*, 1984a) e RF (BREIMAN, 2001). Eles são amplamente utilizados como *baseline* em trabalhos de categorização de texto e outros tipos de problemas binários e multiclasse (LEWIS; RINGUETTE, 1994; JOACHIMS, 1998; YU; XU, 2008; ALMEIDA *et al.*, 2011; WU *et al.*, 2014; APHINYANAPHONGS *et al.*, 2014; ROSSI *et al.*, 2016; ALMEIDA *et al.*, 2016). Além disso, a maioria deles foi listada entre os mais influentes métodos de aprendizado de máquina (WU *et al.*, 2008).

O método proposto nesta tese também foi comparado com uma versão simplificada dele, onde as funções $\hat{S}(d, c_j)$ e $K(t_i)$ foram removidas da Equação 5.2. Esse método simplificado será chamado apenas de MDL no decorrer desta tese. O objetivo é verificar se as funções $\hat{S}(d, c_j)$ e $K(t_i)$ aumentam a capacidade de predição do `MDLText`.

Os métodos M.NB, B.NB, Rocchio, KNN, DT e RF foram implementados em `Python` usando a biblioteca `scikit-learn Library`[3]. Os métodos MDL e `MDLText` foram implementados em `C++`. Já, o método SVM foi implementado usando a biblioteca `LibSVM`[4] para `C++`. Além disso, a função de *kernel* usada no método SVM foi a linear pois, como a dimensionalidade do espaço de atributos é normalmente muito alta, em geral, o método

[3] A biblioteca `scikit-learn Library` está disponível em ⟨http://scikit-learn.org/⟩. Acessado em: 23/01/2017.

[4] A biblioteca `LibSVM` está disponível em ⟨https://www.csie.ntu.edu.tw/~cjlin/libsvm/⟩. Acessado em: 23/01/2017.

SVM com *kernel* linear obtém os melhores resultados, se comparado às outras funções de *kernel* disponíveis (HSU; LIN, 2002).

O desempenho de alguns métodos de categorização de texto pode ser bastante afetado pelo tipo de peso escolhido para representar os documentos. Portanto, para os métodos DT, M.NB, KNN, RF e SVM foi feita uma busca em grade usando validação cruzada *k-fold* estratificada (variação do *k-fold* tradicional que preserva a proporção de exemplos em cada classe) com $k = 5$ para encontrar o melhor esquema de pesos (binário, TF ou TF-IDF). Para os métodos MDL, `MDLText` e Rocchio foi usada apenas a representação TF-IDF, enquanto para o método B.NB foi usada a representação textual binária, pois esses esquemas de representação já são intrínsecos a esses métodos.

O desempenho dos métodos KNN, RF, SVM, MDL e `MDLText` também pode ser bastante afetado pela escolha dos seus parâmetros. Diante disso, foi realizada uma busca em grade para encontrar o melhor valor para o custo, que é o parâmetro de regularização do SVM, para o número de vizinhos do KNN, para o número de árvores usadas pelo RF e para o parâmetro $|\Omega|$ usado pelo MDL e pelo `MDLText`. Para os outros métodos avaliados, foram usados os parâmetros padrões de suas respectivas bibliotecas, com exceção do método Rocchio, onde em vez de usar a medida de distância padrão, que é a distância euclidiana, foi usada a distância de cosseno.

Essas configurações foram adotadas para assegurar que, para cada base de dados, a melhor representação textual e os melhores parâmetros fossem selecionados para cada método. Então, para cada base de dados, o objetivo foi obter o melhor desempenho dos métodos avaliados nos experimentos.

6.1.4 Métodos de aprendizado *online*

O desempenho do `MDLText` também foi comparado aos seguintes métodos bem conhecidos de aprendizado *online*: M.NB (MCCALLUM; NIGAM, 1998), B.NB (MCCALLUM; NIGAM, 1998), Perceptron (FREUND; SCHAPIRE, 1999), SGD(ZHANG, 2004), ROMMA (LI; LONG, 2002) e OGD (ZINKEVICH, 2003).

Assim como nos experimentos com aprendizado *offline*, no cenário *online*, o desempenho do `MDLText` foi comparado ao desempenho do método MDL, que é uma versão simplificada do método apresentado nesta tese, conforme explicado na Seção 6.1.3.

Os métodos M.NB, B.NB, Perceptron e SGD foram implementados em `Python` usando a biblioteca `scikit-learn`. Os métodos OGD e ROMMA foram implementados em `MATLAB` usando funções da biblioteca `LIBOL`[5] (foram usadas as versões multiclasse dos dois métodos que estão disponíveis na biblioteca `LIBOL`) (HOI *et al.*, 2014). Já, o MDL e o `MDLText` foram implementados em `C++`.

[5] A biblioteca `LIBOL` está disponível em: ⟨http://libol.stevenhoi.org/⟩. Acessado em: 19/01/2017.

Uma busca em grade também foi usada para encontrar o melhor esquema de representação dos textos, assim como foi feito nos experimentos com aprendizado *offline*. Para os métodos MDL e `MDLText` foi usado o esquema de representação TF-IDF em todos os experimentos e uma busca em grade foi usada para encontrar o melhor valor do parâmetro $|\Omega|$. Por fim, para o método B.NB, foi usado o esquema de representação binária.

Todos os experimentos foram executados em um computador com processador Intel Core i7 2.93 GHz, memória RAM de 8 GB e sistema operacional Ubuntu 14.04.

6.2 Experimentos e resultados

Nesta seção, são descritos os resultados obtidos nos experimentos com aprendizado *offline* e aprendizado *online*.

6.2.1 Aprendizado *offline*

Antes de comparar o `MDLText` a outros métodos de aprendizado *offline*, foi analisado o seu desempenho usando diferentes técnicas para calcular a pontuação dos termos (Seção 5.1.2).

6.2.1.1 Avaliação do `MDLText` usando diferentes técnicas de pontuação de termos

A Tabela 10 mostra os resultados obtidos pelo `MDLText` para cada uma das 45 bases de dados usando diferentes técnicas para calcular a pontuação dos termos. Os resultados foram obtidos usando validação cruzada 5-*fold* estratificada e calculados pela média da macro F-medida obtida nos experimentos com os cinco *folds*. Para facilitar a comparação dos resultados, os valores são apresentados usando um mapa de calor em tons de cinza onde, para cada base de dados, quanto melhor é a macro F-medida, mais escura é a célula da tabela. Além disso, os valores em negrito indicam o melhor resultado.

Tabela 10 – Macro F-medida obtida pelo método `MDLText` usando diferentes técnicas para o cálculo da pontuação dos termos em um cenário de classificação *offline*. Os valores são apresentados usando um mapa de calor em tons de cinza onde, para cada base de dados, quanto melhor é a macro F-medida, mais escura é a célula da tabela. Os valores em negrito indicam o melhor resultado por base de dados (**Continua**).

	`MDLText` CF	`MDLText` DFS	`MDLText` GSS	`MDLText` IG	`MDLText` NGL	`MDLText` OR	`MDLText` χ^2
20Newsgroups	0,920	0,918	**0,921**	0,920	0,919	0,920	0,919
7Sectors	**0,912**	**0,912**	0,908	0,904	0,906	0,908	0,909

Tabela 10 – Macro F-medida obtida pelo método `MDLText` usando diferentes técnicas para o cálculo da pontuação dos termos em um cenário de classificação *offline*. Os valores são apresentados usando um mapa de calor em tons de cinza onde, para cada base de dados, quanto melhor é a macro F-medida, mais escura é a célula da tabela. Os valores em negrito indicam o melhor resultado por base de dados (**Continuação**).

	`MDLText` CF	`MDLText` DFS	`MDLText` GSS	`MDLText` IG	`MDLText` NGL	`MDLText` OR	`MDLText` χ^2
ACM	0,827	0,823	0,829	0,829	0,824	0,829	**0,830**
CSTR	0,884	0,883	0,887	0,893	0,891	**0,898**	0,871
Dmoz-Business	**0,696**	0,693	0,694	0,690	0,693	0,692	0,690
Dmoz-Computers	**0,710**	0,701	0,701	0,702	0,699	0,703	0,700
Dmoz-Health	**0,828**	0,814	0,819	0,815	0,818	0,820	0,821
Dmoz-Science	**0,740**	0,731	0,736	0,735	0,737	0,735	0,737
Dmoz-Sports	**0,888**	0,875	0,875	0,875	0,878	0,881	0,875
Enron	0,719	0,719	**0,720**	0,719	0,718	0,719	0,719
Fbis	0,786	0,785	0,782	0,789	0,793	0,793	**0,794**
Industry-Sector	**0,813**	0,803	0,806	0,807	0,804	0,806	0,798
Irish economic	0,670	0,672	0,670	0,666	**0,679**	0,664	0,666
La1S	0,875	0,875	0,870	0,870	**0,877**	0,873	0,873
La2S	**0,885**	0,882	0,884	0,880	0,877	0,877	0,882
Latimes	**0,848**	0,846	0,844	0,846	0,847	0,846	0,847
NFS	**0,848**	0,843	0,843	0,839	0,847	0,846	0,843
New3S	0,857	**0,861**	0,857	0,855	0,858	0,860	0,860
Oh0	0,920	**0,923**	0,910	0,912	0,917	0,915	0,909
Oh10	**0,831**	0,812	0,801	0,819	0,796	0,813	0,815
Oh15	0,852	0,845	**0,858**	0,856	**0,858**	0,857	0,856
Oh5	0,911	**0,915**	0,892	0,902	0,892	0,906	0,892
Ohscal	**0,780**	0,775	0,775	0,773	0,774	0,774	0,772
Opinosis	0,669	**0,670**	0,662	0,657	0,662	0,664	0,657
Pubmed-Cancer	**0,926**	0,922	0,910	0,909	0,913	0,915	0,911
Pubmed-Cancer-2000	**0,847**	0,835	0,824	0,824	0,825	0,831	0,824
RCV1	0,926	0,926	**0,927**	0,926	0,925	**0,927**	0,925
Rcv2-Italian	**0,799**	0,791	0,791	0,790	0,795	0,794	0,795
Rcv2-Portuguese	0,789	0,780	0,781	0,789	**0,798**	0,780	0,780
Rcv2-Spanish	0,821	0,832	0,845	0,803	**0,857**	0,842	0,850
Re0	0,811	**0,822**	**0,822**	0,803	0,816	0,799	0,800
Re1	0,815	0,810	**0,816**	0,796	0,802	0,809	0,804
Re8	**0,919**	0,912	0,911	0,908	0,911	0,913	0,907
Reuters	**0,916**	0,911	0,910	0,909	0,912	0,914	0,914
Reviews	**0,902**	0,899	0,896	0,901	**0,902**	0,900	0,901
Techtc300-1092-135724	**0,977**	0,976	0,973	0,969	0,973	0,971	0,974
Techtc300-1092-789236	**0,990**	0,974	0,978	0,978	0,977	0,979	0,977
Tr11	0,849	0,855	0,839	0,841	**0,860**	0,848	0,829

Tabela 10 – Macro F-medida obtida pelo método `MDLText` usando diferentes técnicas para o cálculo da pontuação dos termos em um cenário de classificação *offline*. Os valores são apresentados usando um mapa de calor em tons de cinza onde, para cada base de dados, quanto melhor é a macro F-medida, mais escura é a célula da tabela. Os valores em negrito indicam o melhor resultado por base de dados (**Conclusão**).

	`MDLText` CF	`MDLText` DFS	`MDLText` GSS	`MDLText` IG	`MDLText` NGL	`MDLText` OR	`MDLText` χ^2
Tr12	0,877	**0,903**	0,891	0,883	0,883	0,866	0,870
Tr21	0,865	0,860	**0,880**	0,831	0,859	0,878	**0,880**
Tr23	**0,933**	0,926	0,929	0,918	0,911	0,912	0,898
Tr31	0,823	**0,831**	0,827	0,828	0,828	0,828	0,826
Tr41	0,892	0,924	0,920	**0,933**	0,920	0,922	0,903
Tr45	0,904	0,906	0,908	0,905	0,906	**0,909**	0,894
Trec7-3000	**0,981**	0,976	0,978	0,977	0,975	0,978	0,977

Conforme mostrado na Tabela 10, o `MDLText`, em geral, obteve melhores resultados quando a técnica CF foi utilizada. Porém, na maioria das bases de dados, a diferença entre os valores da macro F-medida obtidos pelas diferentes técnicas foi pequena. Em geral, a diferença entre a melhor macro F-medida e a pior é inferior a 0,02.

Para facilitar a comparação dos resultados e verificar se alguma das técnicas avaliadas foi significativamente melhor que as demais, foi realizada uma análise estatística usando o teste não paramétrico de Friedman, seguindo cuidadosamente a metodologia descrita em Demšar (2006). Para cada técnica, a Figura 10 apresenta o *ranking* médio de cada método de acordo com a macro F-medida, onde *rankings* com médias mais baixas indicam melhor desempenho.

Seja k o número de modelos ou grupos avaliados, q o número de observações e R_j a média dos *rankings* do j-ésimo modelo ou grupo avaliado. O teste de Friedman verifica se a hipótese nula, que afirma que todos os métodos possuem desempenhos equivalentes, pode ser rejeitada. Para isso, é empregada a seguinte equação:

$$\chi_F^2 = \frac{12 \times q}{k \times (k+1)} \times \sum_{j=1}^{k} \left(R_j^2 - \frac{k+1}{2} \right)^2 . \tag{6.1}$$

A hipótese nula é rejeitada se o valor crítico na distribuição χ^2, com $k-1$ graus de liberdade, é menor que χ_F^2 (DEMŠAR, 2006). Para um intervalo de confiança $\alpha = 0{,}05$, $k = 7$ e $q = 45$, o valor crítico é 1,640. Portanto, como $\chi_F^2 = 32{,}036$, a hipótese nula pode ser rejeitada.

Uma vez que a hipótese nula pôde ser rejeitada, foi realizado o teste *post-hoc* para comparar o desempenho individual das técnicas. Para isso, foi usado o teste de

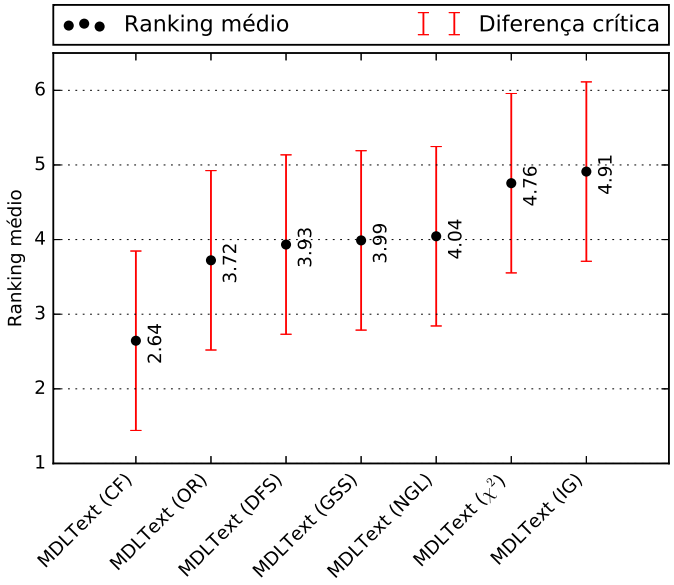


Figura 10 – Média dos *rankings* do `MDLText` usando diferentes técnicas de pontuação de termos em um cenário de classificação *offline* e diferenças críticas calculadas usando o teste *post-hoc* de Bonferroni-Dunn.

Bonferroni-Dunn que afirma que os desempenhos de dois métodos são significativamente divergentes se a diferença entre as médias dos *rankings* deles for maior ou igual a uma diferença crítica calculada pela seguinte equação (GARCÍA *et al.*, 2009; DEMŠAR, 2006):

$$CD = q_\alpha \times \sqrt{\frac{k \times (k+1)}{6 \times q}}, \tag{6.2}$$

onde q_α corresponde ao valor crítico apresentado em Demšar (DEMŠAR, 2006, pág. 12). Para um intervalo de confiança $\alpha = 0{,}05$, a diferença crítica foi igual a 1,202.

Na Figura 10, a diferença crítica é ilustrada pelas barras verticais que estão abaixo e acima dos *rankings* médios. Pode ser observado que o limite superior da barra referente ao `MDLText` (CF) está abaixo do ranking médio das técnicas DFS, GSS, NGL, χ^2 e IG. Isso significa que a diferença entre o *ranking* médio do MDLText (CF) e de cada uma dessas técnicas é maior do que a diferença crítica. Portanto, existe evidência estatística suficiente para poder afirmar que o desempenho do `MDLText` usando a técnica CF foi superior ao desempenho obtido quando as técnicas DFS, GSS, NGL, χ^2 e IG foram usadas. Porém, não existe evidência estatística para afirmar que o desempenho obtido usando a técnica CF foi superior ao desempenho obtido com a técnica OR. Apesar disso, como o `MDLText` obteve o menor *ranking* médio quando foi usada a técnica CF, ela foi utilizada em todos os experimentos posteriores.

6.2.1.2 Comparação do `MDLText` com outros métodos de classificação

A Tabela 11 apresenta os resultados obtidos pelos métodos para cada uma das 45 bases de dados. Os resultados foram calculados usando a macro F-medida e validação cruzada 5-*fold* estratificada. Os valores também são apresentados usando um mapa de calor em tons de cinza e os valores em negrito indicam o melhor resultado.

Tabela 11 – Macro F-medida obtida por cada método na classificação dos textos de cada base de dados usando validação cruzada 5-*fold* estratificada (**Continua**).

	B.NB	DT	KNN	M.NB	MDL	`MDLText`	RF	Rocchio	SVM
20Newsgroups	0,813	0,686	0,864	0,895	0,913	0,920	0,860	0,867	**0,927**
7Sectors	0,548	0,810	0,920	0,855	0,889	0,912	0,890	0,885	**0,957**
ACM	0,653	0,643	0,776	0,774	0,828	0,827	0,843	0,814	**0,883**
CSTR	0,722	0,668	0,871	**0,885**	0,852	0,884	0,763	0,883	0,863
Dmoz-Business	0,665	0,488	0,683	0,698	0,648	0,696	0,632	0,649	**0,703**
Dmoz-Computers	0,682	0,529	0,685	0,705	0,670	**0,710**	0,656	0,667	**0,710**
Dmoz-Health	0,795	0,718	0,813	0,821	0,793	0,828	0,799	0,793	**0,850**
Dmoz-Science	0,700	0,557	0,695	0,737	0,696	**0,740**	0,676	0,707	0,732
Dmoz-Sports	0,830	0,816	0,835	0,851	0,811	0,888	0,882	0,854	**0,908**
Enron	0,523	0,578	0,692	0,709	0,722	0,719	0,681	0,689	**0,734**
Fbis	0,609	0,644	0,713	0,739	0,739	0,786	0,783	0,789	**0,858**
Industry-Sector	0,462	0,506	0,695	0,740	0,778	0,813	0,693	0,746	**0,903**
Irish economic	0,659	0,503	0,611	0,653	0,656	0,670	0,647	0,672	**0,676**
La1S	0,763	0,728	0,825	0,857	0,878	0,875	0,843	0,864	**0,899**
La2S	0,782	0,753	0,858	0,876	0,873	0,885	0,856	0,869	**0,904**
Latimes	0,792	0,697	0,805	0,828	0,841	0,848	0,795	0,824	**0,849**
NFS	0,655	0,698	0,782	0,827	0,815	**0,848**	0,780	0,782	0,840
New3S	0,473	0,703	0,797	0,802	0,816	0,857	0,854	0,852	**0,909**
Oh0	0,711	0,793	0,888	0,873	0,860	**0,920**	0,895	0,919	0,902
Oh10	0,667	0,721	0,776	0,765	0,771	**0,831**	0,820	0,812	0,813
Oh15	0,677	0,739	0,813	0,822	0,784	0,852	0,840	**0,864**	0,851
Oh5	0,737	0,828	0,853	0,863	0,806	**0,911**	0,891	0,908	0,910
Ohscal	0,720	0,676	0,724	0,737	0,723	0,780	0,793	0,779	**0,801**
Opinosis	0,204	0,577	0,581	0,578	0,522	0,669	0,617	**0,673**	0,626
Pubmed-Cancer	0,667	0,936	0,598	0,762	0,684	0,926	0,841	0,903	**0,944**
Pubmed-Cancer-2000	0,645	0,862	0,598	0,730	0,658	0,847	0,850	0,831	**0,872**
RCV1	0,835	0,858	0,922	0,928	0,929	0,926	0,926	0,891	**0,963**
Rcv2-Italian	0,739	0,754	0,845	0,804	0,809	0,799	0,820	0,770	**0,868**
Rcv2-Portuguese	0,694	0,680	0,795	0,793	0,782	0,789	0,723	0,797	**0,838**
Rcv2-Spanish	0,678	0,725	0,828	0,705	0,841	0,821	0,844	0,788	**0,875**
Re0	0,280	0,619	0,760	0,712	0,711	0,811	0,714	0,823	**0,865**
Re1	0,237	0,742	0,718	0,693	0,671	**0,815**	0,682	0,807	0,780
Re8	0,604	0,822	0,865	0,907	0,884	0,919	0,850	0,883	**0,943**
Reuters	0,472	0,845	0,889	0,907	0,884	0,916	0,874	0,874	**0,953**
Reviews	0,769	0,840	0,905	0,907	0,894	0,902	0,884	0,894	**0,933**

Tabela 11 – Macro F-medida obtida por cada método na classificação dos textos de cada base de dados usando validação cruzada 5-*fold* estratificada (**Conclusão**).

	B.NB	DT	KNN	M.NB	MDL	`MDLText`	RF	Rocchio	SVM
Techtc300-1092-135724	0,896	0,954	0,953	0,960	**0,982**	0,977	0,948	0,968	0,975
Techtc300-1092-789236	0,892	0,972	0,976	0,976	0,985	**0,990**	0,968	0,960	0,977
Tr11	0,338	0,684	0,646	0,698	0,754	**0,849**	0,675	0,789	0,749
Tr12	0,481	0,794	0,780	0,806	0,801	0,877	0,774	**0,896**	0,854
Tr21	0,355	0,649	0,754	0,582	0,651	0,865	0,442	**0,886**	0,648
Tr23	0,273	0,843	0,827	0,794	0,871	**0,933**	0,650	0,931	0,888
Tr31	0,531	0,774	0,783	0,784	0,795	0,823	0,818	0,822	**0,833**
Tr41	0,546	0,861	0,918	0,908	0,880	0,892	0,833	0,905	**0,934**
Tr45	0,607	0,839	0,821	0,807	0,854	**0,904**	0,836	0,880	0,871
Trec7-3000	0,698	0,970	0,977	0,973	0,975	0,981	0,990	0,954	**0,995**

Os resultados indicam que o método SVM obteve os melhores resultados na maioria das bases de dados. Porém, o `MDLText` obteve um desempenho superior a todos os outros métodos, com exceção do SVM, em 21 bases de dados. Além disso, o `MDLText` foi bastante superior ao MDL e foi o método mais consistente como o segundo melhor método de classificação.

As diferenças entre a macro F-medida do MDL e do `MDLText` foram maiores nas bases de dados com alto desbalanceamento entre as classes. Por exemplo, nas bases de dados PubMed-Cancer, Tr21, PubMed-Cancer-2000, Opinosis e Re1, a diferença absoluta entre a macro F-medida dos dois métodos foi 0,242, 0,214, 0,189, 0,147 e 0,144, respectivamente. Para verificar se tais diferenças foram causadas pelo fato de que o MDL tem desempenho ruim na predição das classes raras, foi calculada a micro F-medida (para maiores detalhes sobre esse medida de desempenho, consulte a Seção 2.4). As diferenças entre os valores da micro F-medida obtidos pelo MDL e `MDLText` foram, respectivamente, 0,152, 0,130, 0,167, 0,135 e 0,076. Dado que a micro F-medida é dominada pela classe majoritária, tais diferenças indicam que o método MDL foi mais negativamente afetado pelo desbalanceamento entre as classes do que o `MDLText`.

Também pode ser observado que foram em experimentos com bases de dados altamente desbalanceadas que o `MDLText` obteve os piores resultados em comparação ao SVM. Por exemplo, nas base de dados Industry-Sector, RCV2-Italian, RCV2-Spanish e RCV2-Portuguese a diferença absoluta entre a macro F-medida dos dois métodos foi 0,090, 0,069, 0,054 e 0,049, respectivamente. Portanto, pode-se concluir que o método proposto é mais negativamente influenciado pelo desbalanceamento das classes do que o SVM. Porém, o `MDLText` foi mais robusto ao problema de desbalanceamento do que os métodos B.NB, DT, KNN, M.NB, RF e Rocchio em várias classes, tais como a Enron, Fbis, Industry-Sector, Latimes, NFS, New3S, Ohscal, Re1, Re8, Reuters, Tr11, Tr23, Tr31

e Tr45.

Para certificar que os resultados reportados na Tabela 11 não foram obtidos ao acaso, foi realizada uma análise estatística usando o teste não paramétrico de Friedman. A Figura 11 apresenta o *ranking* médio de cada método avaliado.

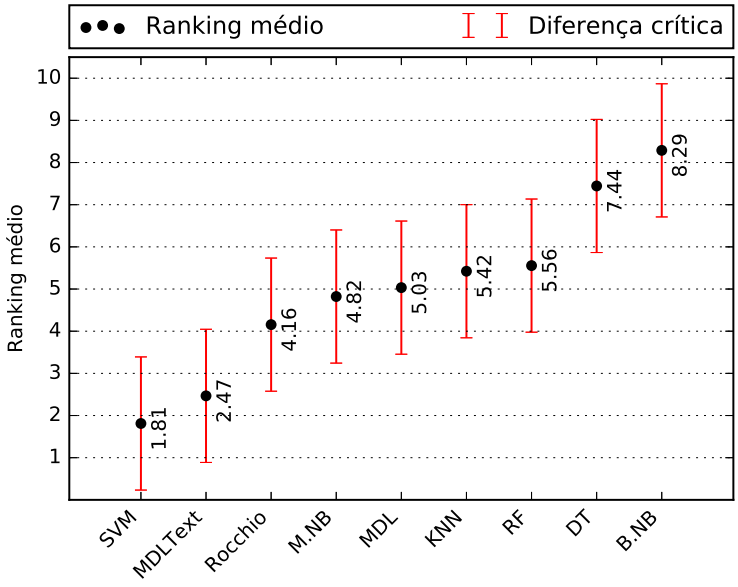


Figura 11 – *Ranking* médio de cada método em um cenário de classificação *offline* e diferenças críticas calculadas usando o teste *post-hoc* de Bonferroni–Dunn.

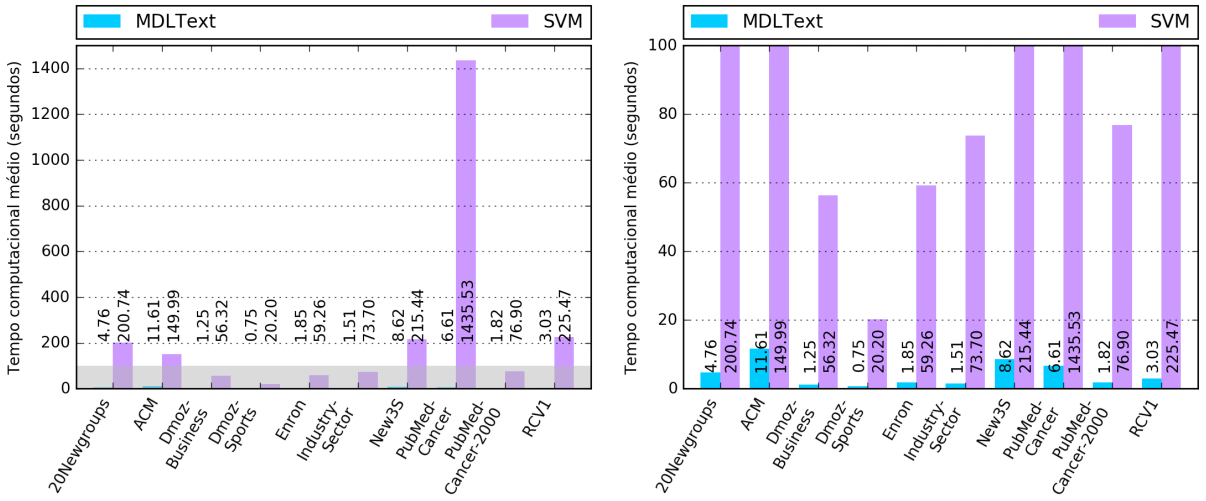
Para um intervalo de confiança $\alpha = 0{,}05$, $k = 9$ e $q = 45$, o valor crítico calculado pelo teste de Friedman foi 2,730. Portanto, como $\chi_F^2 = 207{,}670$, a hipótese nula, que afirma que todos os métodos são equivalentes, foi rejeitada.

Uma vez que a hipótese nula pôde ser rejeitada, foi feita uma comparação para-a-par entre os métodos avaliados usando o teste *post-hoc* de Bonferroni-Dunn. Para um intervalo de confiança $\alpha = 0{,}05$, $k = 9$ e $q = 45$, a diferença crítica foi 1,579. Portanto, não há evidência estatística suficiente para afirmar que o desempenho do método SVM foi superior ao do `MDLText`. Porém, existe evidência estatística suficiente para seguramente afirmar que o desempenho do método `MDLText` foi significativamente superior ao desempenho dos métodos MDL, M.NB, B.NB, k-NN, DT, RF e Rocchio

Apesar de ser estatisticamente equivalente ao SVM, o `MDLText` possui características que o tornam vantajoso em aplicações que envolvem cenários reais e problemas de categorização de texto de grande escala. Algumas dessas características incluem seu rápido processo de treinamento e sua capacidade de aprendizado de forma incremental.

Na Seção 5.1.1, já foi mostrado que o `MDLText` tem complexidade linear tanto no processo de treinamento quanto na predição, o que o torna tão eficiente quanto qualquer outro método de complexidade linear. Apesar disso, para oferecer uma melhor análise, a Figura 12 compara o tempo computacional médio, em segundos, dos métodos `MDLText`

e SVM no treinamento e classificação das dez maiores bases de dados analisadas nesse trabalho. Em média, **o SVM foi 56 vezes mais lento** que o `MDLText`.



(a) (b)

Figura 12 – Média do tempo computacional (em segundos) demandado no treinamento e classificação das dez maiores bases de dados. A Figura 12b ilustra a região cinza da Figura 12a.

Além de apresentar um custo computacional bem maior que o método apresentado nesta tese, o SVM tradicional não suporta treinamento incremental. Portanto, em um problema que necessite de aprendizado *online*, o estágio de treinamento do SVM deve ser refeito de tempo em tempo, o que pode demandar um tempo impraticável quando há um grande aumento no número de exemplos e atributos. Por outro lado, o `MDLText` é naturalmente multiclasse, rápido e pode ser treinado incrementalmente, o que torna possível sua aplicação em problemas reais, *online* e dinâmicos.

6.2.1.3 Seleção de atributos

Nesta seção, foi avaliado se a seleção de atributos impacta o desempenho dos métodos de classificação. O objetivo principal é verificar se o `MDLText`, que usa uma função de pontuação para os termos, é afetado negativamente se o número de termos for reduzido.

Foram realizados experimentos com um método de seleção baseado em filtro, pois, conforme foi mencionado na Seção 3.1.4, métodos baseados em filtros geralmente são mais simples e escaláveis do que métodos *wrappers* e *embedded* e, ainda, são independentes do método de classificação. O método utilizado foi o IG que, de acordo com Yang e Pedersen (1997), é o mais efetivo. Além disso, ele é um dos métodos de seleção de atributos mais empregados na literatura (LEWIS; RINGUETTE, 1994; ROGATI; YANG, 2002; SAIF *et al.*, 2012; KILINÇ *et al.*, 2015).

Nas Figuras 13 a 22, são apresentados os resultados obtidos na classificação das dez bases de dados com maior número de atributos. O IG foi usado para selecionar 20%,

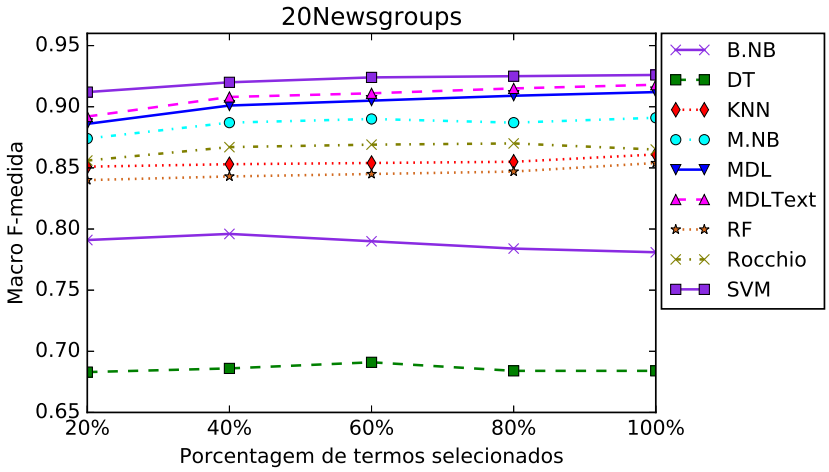


Figura 13 – Macro F-medida obtida por cada método após a seleção de atributos – base de dados 20Newsgroups.

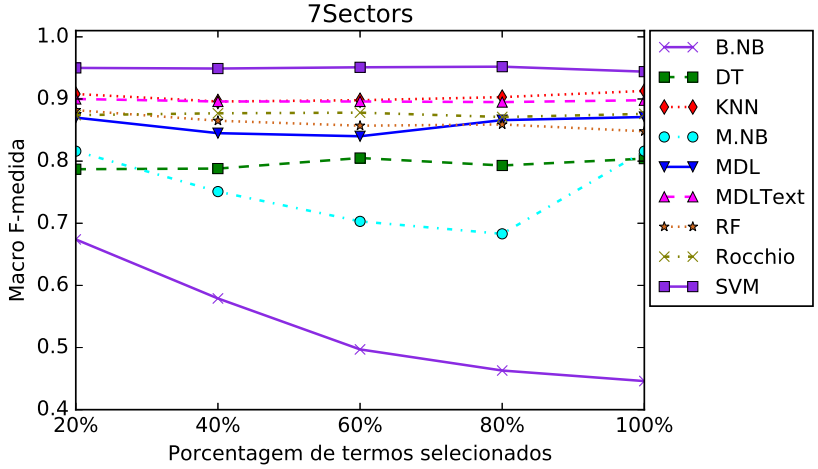


Figura 14 – Macro F-medida obtida por cada método após a seleção de atributos – base de dados 7Sectors.

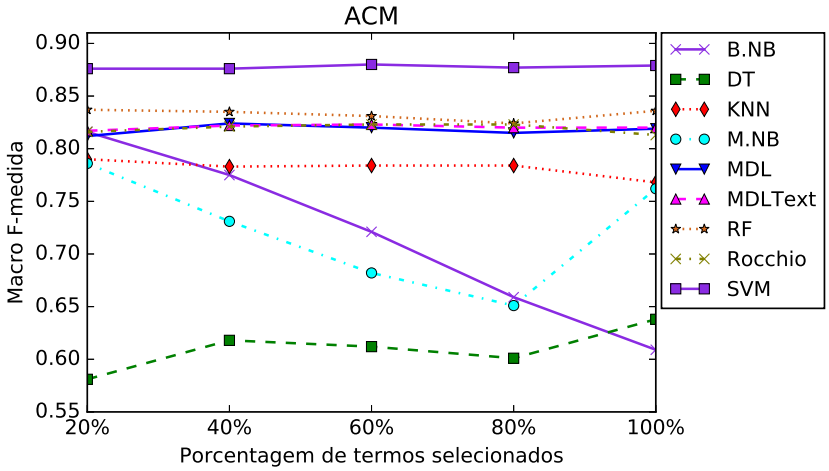


Figura 15 – Macro F-medida obtida por cada método após a seleção de atributos – base de dados ACM.

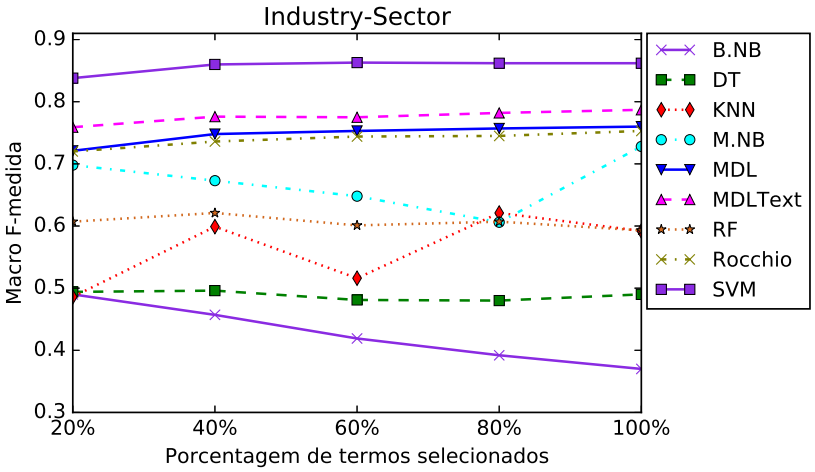


Figura 16 – Macro F-medida obtida por cada método após a seleção de atributos – base de dados Industry-Sector.

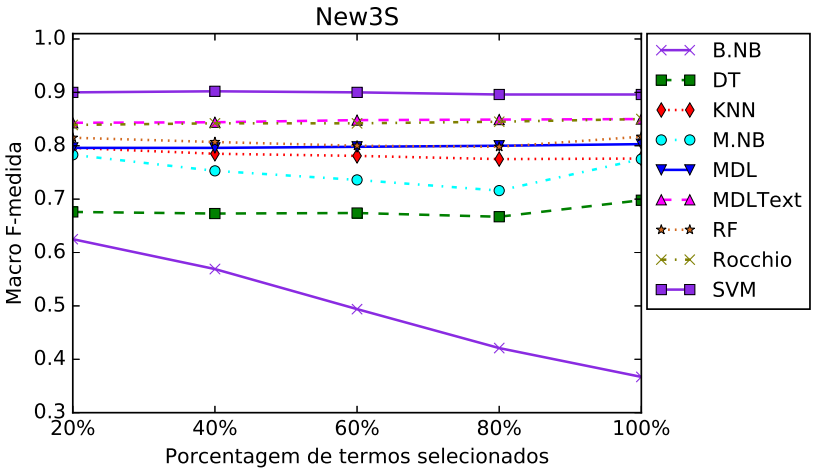


Figura 17 – Macro F-medida obtida por cada método após a seleção de atributos – base de dados New3S.

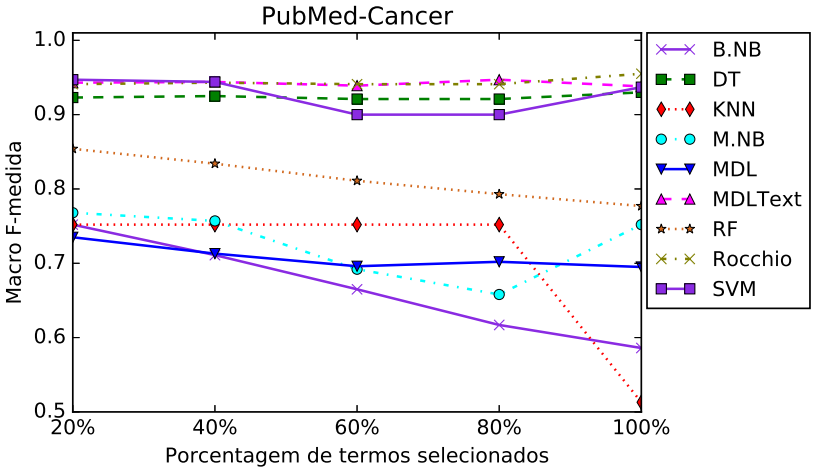


Figura 18 – Macro F-medida obtida por cada método após a seleção de atributos – base de dados PubMed-Cancer.

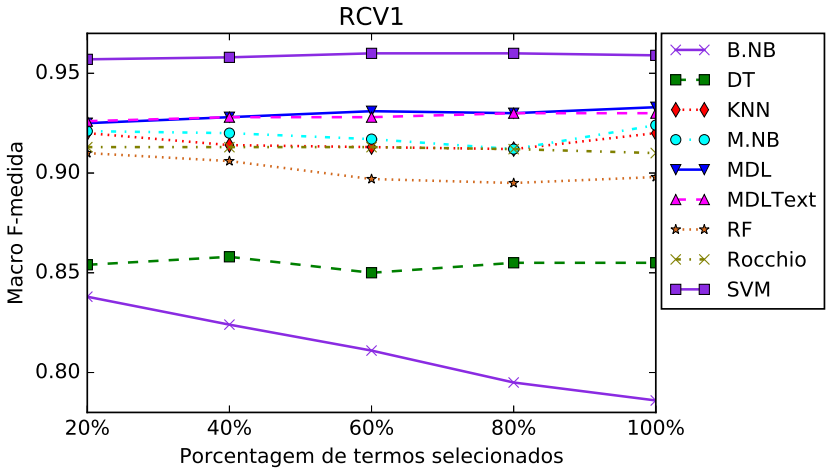


Figura 19 – Macro F-medida obtida por cada método após a seleção de atributos – base de dados RCV1.

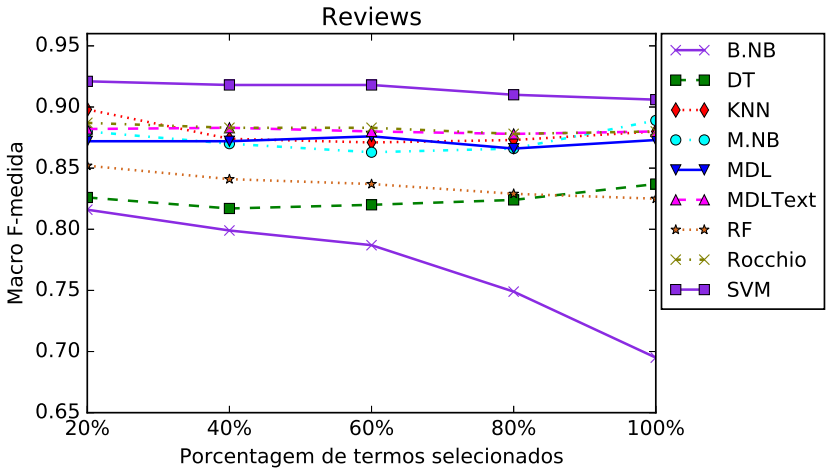


Figura 20 – Macro F-medida obtida por cada método após a seleção de atributos – base de dados Reviews.

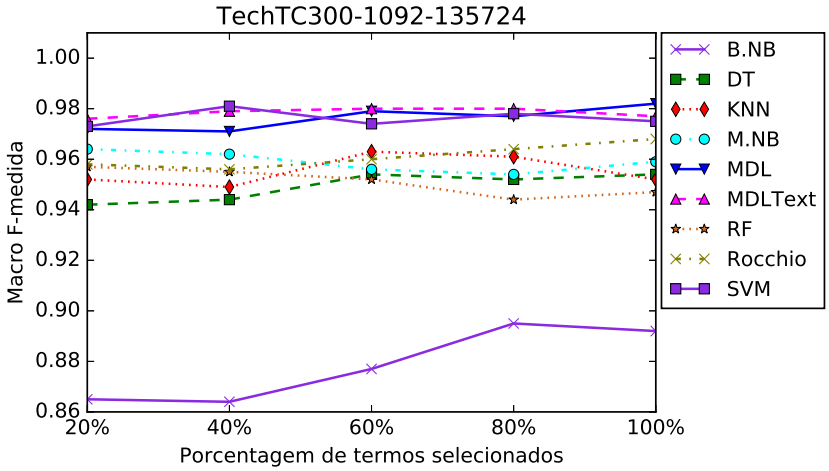


Figura 21 – Macro F-medida obtida por cada método após a seleção de atributos – base de dados TechTC300-1092-135724.

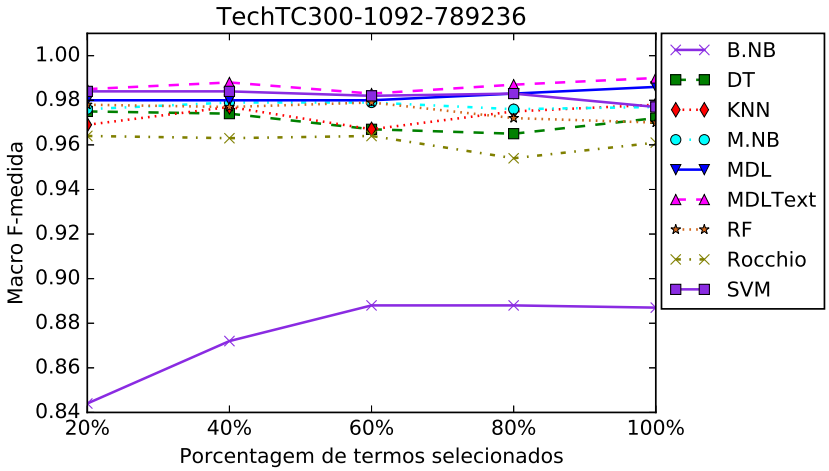


Figura 22 – Macro F-medida obtida por cada método após a seleção de atributos – base de dados TechTC300-1092-789236.

40%, 60% e 80% dos melhores atributos. Os resultados foram comparados com aqueles obtidos sem a aplicação de seleção de atributos (100%). Os experimentos, para cada porcentagem de seleção de atributos, seguiram o mesmo procedimento descrito na Seção 6.1. Os resultados mostrados nos gráficos foram calculados usando a macro F-medida e validação cruzada 5-*fold* estratificada.

Conforme pode ser observado nas Figuras 13 a 22, o desempenho do `MDLText` foi pouco influenciado pela seleção de atributos. A maior variação ocorreu nos experimentos com a base de dados 20Newsgroups, porém a diferença entre o melhor e o pior resultado foi menor que 0,03. Também, é possível observar que o B.NB foi o único método que obteve uma melhoria significativa no desempenho após a seleção de atributos. O KNN também apresentou resultado bastante superior com a seleção de atributos nos experimentos com a base de dados PubMed-Cancer. Os demais métodos, de maneira geral, foram pouco impactados com a seleção de atributos.

6.2.2 Aprendizado *online*

Nestes experimentos, foi considerado o seguinte cenário: um pequeno número de amostras (20% dos documentos de cada classe) é usado na busca em grade realizada para definir o melhor esquema de pesos. Depois elas são disponibilizadas para o treinamento inicial dos métodos. Em seguida, um exemplo por vez é apresentado ao classificador para que este faça sua predição. Após isso, o classificador recebe um *feedback* e, caso o erro seja maior que zero, o modelo de predição é atualizado com a classe correta. Todo o processo é descrito no Algoritmo 6.1, onde foi considerado que os documentos de entrada são representados por um modelo espaço-vetorial usando o esquema de pesos TF. Caso o esquema de representação de pesos escolhido na busca em grade seja o TF-IDF, o processo de conversão também é aplicado incrementalmente, pois esse tipo de representação precisa

de informações sobre os documentos apresentados no processo de treinamento, e nestes experimentos, deseja-se reproduzir um cenário real e dinâmico, em vez de um estático.

Pseudocódigo 6.1 *Framework* para o aprendizado *online*

função APRENDIZADO_ONLINE($\mathcal{D}$, *lista_de_esquema_de_pesos*)
    $\mathcal{D}_{treino} \leftarrow$ selecione aleatoriamente 20% dos exemplos de cada classe contidos em $\mathcal{D}$
    $\mathcal{D}_{teste} \leftarrow$ selecione o restante dos documentos contidos em $\mathcal{D}$

    se quantidade de esquemas em *lista_de_esquema_de_pesos* > 1 **então**
        esquema_de_pesos $\leftarrow$ use uma busca em grade com os exemplos contidos em $\mathcal{D}_{treino}$ para selecionar o melhor esquema em *lista_de_esquema_de_pesos*
    senão
        esquema_de_pesos $\leftarrow$ *lista_de_esquema_de_pesos*
    fim se

    se *esquema_de_pesos* = "TF-IDF" **então**
        $DF \leftarrow$ quantidade de documentos de treinamento em que cada termo aparece
        $n_{treino} \leftarrow$ quantidade de documentos em $\mathcal{D}_{treino}$
        $\mathcal{D}_{treino} \leftarrow$ converte os documentos de $\mathcal{D}_{treino}$ para TF-IDF usando DF e n_{treino}
    senão se *esquema_de_pesos* = "binário" **então**
        $\mathcal{D}_{treino} \leftarrow$ converte os documentos de $\mathcal{D}_{treino}$ para binário
    fim se

    modelo $\leftarrow$ inicializa o modelo
    modelo $\leftarrow$ atualiza o modelo usando $\mathcal{D}_{treino}$
    $i \leftarrow 1$
    para cada documento d em $\mathcal{D}_{teste}$ **faça**

        se *esquema_de_pesos* = "TF-IDF" **então**
            $\hat{d} \leftarrow$ cópia de d antes de convertê-lo para TF-IDF
            $d \leftarrow$ converte d para TF-IDF usando DF e n_{treino}
        senão se *esquema_de_pesos* = "binário" **então**
            $d \leftarrow$ converte d para binário
        fim se

        $\hat{c}_i \leftarrow$ o classificador faz a predição da classe do documento
        $c_i \leftarrow$ classe verdadeira do documento
        erro $\leftarrow$ o classificador calcula o erro de predição
        se *erro* > 0 **então**

            se *esquema_de_pesos* = "TF-IDF" **então**
                $DF \leftarrow$ atualiza DF
                $n_{treino} \leftarrow n_{treino} + 1$
                $d \leftarrow$ converte $\hat{d}$ para TF-IDF usando DF e n_{treino}
            fim se

            modelo $\leftarrow$ atualiza o modelo usando a classe c_i
        fim se
        $i \leftarrow i + 1$
    fim para
    retorna $\hat{c}$ e c
fim função

Antes de comparar o `MDLText` com outros métodos de aprendizado *online*, foi analisado o seu desempenho usando as diferentes técnicas para o cálculo da pontuação dos termos.

6.2.2.1 Avaliação do MDLText usando diferentes técnicas de pontuação dos termos

A Tabela 12 mostra os resultados obtidos pelo MDLText para cada uma das 45 bases de dados usando diferentes técnicas para o cálculo da pontuação dos termos. Os valores mostrados na tabela são as médias da macro F-medida obtidas em dez rodadas do experimento descrito no Algoritmo 6.1.

Tabela 12 – Média da macro F-medida obtida pelo MDLText usando diferentes técnicas para o cálculo da pontuação dos termos em um cenário de classificação *online* (**Continua**).

	MDLText CF	MDLText DFS	MDLText GSS	MDLText IG	MDLText NGL	MDLText OR	MDLText χ^2
20Newsgroups	0,896	0,894	0,894	0,895	**0,897**	0,896	0,896
7Sectors	0,897	0,898	**0,899**	0,893	0,895	0,894	0,894
ACM	0,806	**0,808**	0,805	0,805	0,803	0,806	0,805
CSTR	0,831	0,840	**0,845**	0,828	0,839	0,836	0,841
Dmoz-Business	**0,668**	0,664	0,665	0,666	0,665	0,664	0,663
Dmoz-Computers	**0,677**	0,673	0,671	0,673	0,675	0,675	0,671
Dmoz-Health	**0,808**	0,803	0,800	0,801	0,805	0,803	0,802
Dmoz-Science	**0,694**	0,686	0,692	0,687	0,689	0,691	0,685
Dmoz-Sports	**0,873**	0,867	0,865	0,864	0,866	0,868	0,866
Enron	**0,721**	0,719	0,717	0,719	0,718	0,719	0,717
Fbis	**0,790**	0,784	0,787	0,778	0,782	0,788	0,785
Industry-Sector	**0,782**	0,778	0,781	0,780	0,781	0,780	0,776
Irish economic	0,645	0,638	0,637	0,642	0,645	0,641	**0,647**
La1S	0,873	**0,874**	0,871	0,870	0,871	0,869	0,871
La2S	0,879	0,879	0,877	**0,881**	0,879	0,876	0,879
Latimes	0,836	0,836	0,833	0,834	0,834	0,835	**0,837**
NFS	**0,810**	0,804	0,806	0,806	0,806	0,808	0,805
New3S	**0,858**	0,857	0,857	0,857	0,855	**0,858**	**0,858**
Oh0	0,885	0,882	**0,887**	0,884	0,882	0,884	0,880
Oh10	**0,796**	0,784	0,778	0,783	0,789	0,786	0,785
Oh15	**0,826**	0,824	0,818	0,815	0,822	**0,826**	0,824
Oh5	0,867	**0,874**	0,871	0,860	0,869	0,868	0,864
Ohscal	**0,776**	0,772	0,771	0,772	0,772	0,775	0,773
Opinosis	**0,643**	0,636	0,629	0,631	0,633	0,636	0,630
Pubmed-Cancer	**0,924**	**0,924**	0,913	0,914	0,917	0,918	0,914
Pubmed-Cancer-2000	**0,836**	0,830	0,824	0,825	0,825	0,829	0,826
RCV1	**0,926**	**0,926**	0,925	0,925	0,925	**0,926**	**0,926**
Rcv2-Italian	0,821	0,823	0,823	0,823	**0,824**	0,822	**0,824**
Rcv2-Portuguese	0,806	0,808	0,801	0,811	0,811	0,809	**0,812**
Rcv2-Spanish	0,822	0,823	**0,838**	0,827	0,813	0,826	0,823
Re0	0,787	0,780	0,791	0,781	**0,793**	0,784	0,770
Re1	0,753	0,752	**0,757**	0,746	0,748	0,753	0,745
Re8	**0,914**	0,905	0,905	0,910	0,909	0,907	0,908

Tabela 12 – Média da macro F-medida obtida pelo `MDLText` usando diferentes técnicas para o cálculo da pontuação dos termos em um cenário de classificação *online* (**Conclusão**).

	`MDLText` CF	`MDLText` DFS	`MDLText` GSS	`MDLText` IG	`MDLText` NGL	`MDLText` OR	`MDLText` χ^2
Reuters	**0,917**	0,916	**0,917**	0,916	0,915	0,914	0,914
Reviews	0,910	0,911	0,912	0,910	0,913	**0,914**	0,909
Techtc300-1092-135724	**0,972**	0,970	0,967	0,968	**0,972**	0,971	**0,972**
Techtc300-1092-789236	**0,982**	0,975	0,976	0,978	0,976	0,977	0,975
Tr11	0,795	0,786	0,796	**0,799**	0,792	0,791	0,795
Tr12	0,839	0,843	**0,848**	0,842	0,841	0,841	0,836
Tr21	0,856	0,822	0,856	0,857	0,825	0,827	**0,858**
Tr23	**0,883**	0,869	0,879	0,872	0,862	0,867	0,866
Tr31	**0,818**	**0,818**	0,815	0,816	0,811	0,817	0,813
Tr41	0,922	**0,927**	**0,927**	0,926	0,919	0,919	0,920
Tr45	**0,904**	0,900	0,894	0,898	0,896	0,895	0,899
Trec7-3000	**0,984**	**0,984**	0,983	0,983	0,983	0,983	**0,984**

Os resultados indicam que, assim como nos experimentos com aprendizado *offline*, para a maioria das bases de dados, o `MDLText` obteve melhor desempenho quando a técnica CF foi usada para calcular a pontuação dos termos. Além disso, o desvio padrão entre os resultados obtidos pelo `MDLText` usando as diferentes técnicas foi pequeno.

Para verificar se alguma das técnicas avaliadas foi significativamente melhor que as demais, foi realizada uma análise estatística usando o teste não paramétrico de Friedman, considerando os *rankings* médios mostrados na Figura 23.

Para um intervalo de confiança $\alpha = 0{,}05$, $k = 7$ e $q = 45$, o valor crítico calculado pelo teste de Friedman foi 1,640. Portanto, como $\chi^2_F = 30{,}460$, a hipótese nula de igualdade entre as técnicas pode ser rejeitada.

Diante disso, foi feita uma comparação par-a-par usando o teste *post-hoc* de Bonferroni-Dunn. Para um intervalo de confiança $\alpha = 0{,}05$, $k = 7$ e $q = 45$, a diferença crítica foi igual a 1,202. Portanto, pode-se afirmar com segurança que o desempenho do `MDLText` usando a técnica CF foi estatisticamente superior ao desempenho obtido com as demais técnicas. Sendo assim, ela foi usada em todos os experimentos subsequentes.

6.2.2.2 Comparação do `MDLText` com outros métodos de classificação

A Tabela 13 apresenta os resultados de cada método para cada uma das 45 bases de dados. Os valores mostrados são a média da macro F-medida obtida por cada método de classificação em dez rodadas do experimento descrito no Algoritmo 6.1.

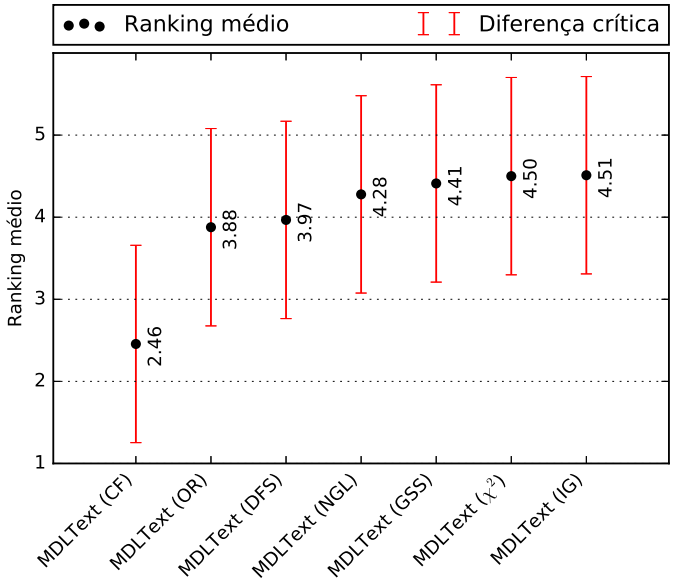


Figura 23 – Média dos *rankings* do `MDLText` usando diferentes técnicas de pontuação de termos em um cenário de classificação *online* e diferenças críticas calculadas usando o teste *post-hoc* de Bonferroni–Dunn.

Tabela 13 – Média da macro F-medida obtida por cada método na classificação *online* em cada base de dados (**Continua**).

	B.NB	M.NB	MDL	`MDLText`	OGD	Perceptron	ROMMA	SGD
20Newsgroups	0,807	0,884	0,876	**0,896**	0,816	0,831	0,801	0,878
7Sectors	0,718	0,827	0,846	0,897	0,845	0,858	0,723	**0,898**
ACM	0,655	0,725	0,791	**0,806**	0,726	0,686	0,556	0,751
CSTR	0,673	**0,833**	0,779	0,831	0,787	0,712	0,628	0,761
Dmoz-Business	0,612	**0,670**	0,607	0,668	0,623	0,550	0,571	0,605
Dmoz-Computers	0,628	0,673	0,623	**0,677**	0,635	0,570	0,122	0,612
Dmoz-Health	0,773	0,806	0,756	**0,808**	0,764	0,724	0,761	0,773
Dmoz-Science	0,650	**0,697**	0,629	0,694	0,660	0,589	0,608	0,641
Dmoz-Sports	0,803	0,850	0,784	**0,873**	0,854	0,820	0,839	0,863
Enron	0,554	0,693	0,706	**0,721**	0,633	0,660	0,005	0,694
Fbis	0,599	0,725	0,721	**0,790**	0,636	0,705	0,639	0,743
Industry-Sector	0,479	0,721	0,719	0,782	0,607	0,756	0,016	**0,802**
Irish economic	0,640	0,638	0,612	**0,645**	0,601	0,560	0,533	0,596
La1S	0,805	0,844	0,854	**0,873**	0,814	0,795	0,801	0,817
La2S	0,814	0,863	0,860	**0,879**	0,829	0,803	0,768	0,849
Latimes	0,780	0,812	0,805	**0,836**	0,764	0,737	0,741	0,780
NFS	0,634	0,789	0,761	**0,810**	0,754	0,704	0,742	0,760
New3S	0,501	0,775	0,799	**0,858**	0,797	0,802	0,785	0,854
Oh0	0,727	0,841	0,802	**0,885**	0,815	0,780	0,768	0,843
Oh10	0,678	0,745	0,713	**0,796**	0,733	0,662	0,680	0,715
Oh15	0,668	0,775	0,714	**0,826**	0,764	0,686	0,650	0,764
Oh5	0,730	0,826	0,760	**0,867**	0,815	0,756	0,737	0,820

Tabela 13 – Média da macro F-medida obtida por cada método na classificação *online* em cada base de dados (**Conclusão**).

	B.NB	M.NB	MDL	`MDLText`	OGD	Perceptron	ROMMA	SGD
Ohscal	0,714	0,729	0,705	**0,776**	0,769	0,698	0,692	0,746
Opinosis	0,231	0,579	0,493	**0,643**	0,610	0,501	0,591	0,553
Pubmed-Cancer	0,666	0,786	0,703	0,924	0,812	0,891	0,819	**0,947**
Pubmed-Cancer-2000	0,666	0,743	0,659	0,836	**0,841**	0,790	0,016	0,823
RCV1	0,849	0,926	0,923	0,926	0,880	0,921	0,892	**0,945**
Rcv2-Italian	0,718	0,817	0,815	0,821	0,690	0,805	0,816	**0,828**
Rcv2-Portuguese	0,625	0,802	0,791	**0,806**	0,485	0,745	0,739	0,772
Rcv2-Spanish	0,655	0,726	0,807	**0,822**	0,526	0,757	0,747	0,780
Re0	0,308	0,710	0,654	**0,787**	0,680	0,654	0,658	0,746
Re1	0,248	0,657	0,593	**0,753**	0,636	0,659	0,648	0,716
Re8	0,584	0,891	0,850	0,914	0,745	0,859	0,885	**0,916**
Reuters	0,458	0,875	0,861	0,917	0,799	0,877	0,898	**0,930**
Reviews	0,796	0,890	0,877	**0,910**	0,843	0,870	0,799	0,906
Techtc300-1092-135724	0,939	0,957	0,965	**0,972**	0,917	0,942	0,798	0,946
Techtc300-1092-789236	0,962	0,974	0,976	**0,982**	0,933	0,957	0,801	0,962
Tr11	0,439	0,676	0,647	**0,795**	0,613	0,650	0,605	0,753
Tr12	0,547	0,750	0,731	**0,839**	0,567	0,762	0,648	0,813
Tr21	0,311	0,679	0,503	**0,856**	0,294	0,775	0,725	0,800
Tr23	0,316	0,750	0,681	**0,883**	0,547	0,758	0,600	0,850
Tr31	0,580	0,782	0,776	**0,818**	0,689	0,782	0,708	0,817
Tr41	0,549	0,872	0,853	**0,922**	0,803	0,860	0,698	0,909
Tr45	0,642	0,798	0,790	**0,904**	0,795	0,846	0,728	0,897
Trec7-3000	0,903	0,978	0,980	0,984	0,909	0,978	0,388	**0,989**

O `MDLText` também obteve bons resultados nos experimentos com aprendizado *online*. Para a maioria das bases de dados (33 de 45), ele foi superior a todos os demais métodos comparados. Além disso, ele obteve melhores resultados que o MDL em todas as bases de dados.

Foi realizada uma análise estatística baseada na média dos *rankings* de cada método de classificação (Figura 24). De acordo com o teste não-paramétrico de Friedman, com $\alpha = 0{,}05$, $k = 8$ e $q = 45$, o valor crítico foi 2,170. Dado que $\chi_F^2 = 185{,}683$, foi rejeitada a hipótese nula de que todos os métodos comparados nos experimentos com aprendizado *online* são equivalentes. Também, foi feita uma comparação par-a-par usando o teste *post-hoc* de Bonferroni–Dun. Para um intervalo de confiança $\alpha = 0{,}05$, a diferença crítica foi igual a 1,389 e, portanto, pode-se afirmar com segurança que o desempenho do `MDLText` foi significativamente melhor que o desempenho de todos os demais métodos comparados.

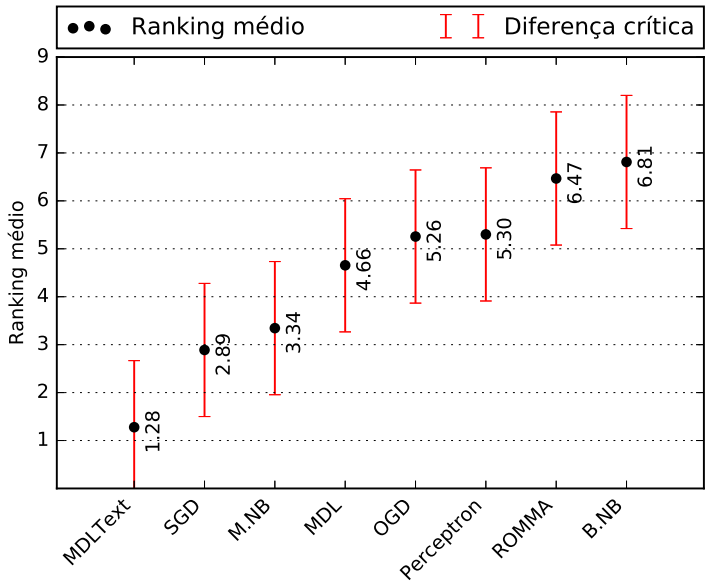


Figura 24 – *Ranking* médio de cada método em um cenário de classificação *online* e diferenças críticas calculadas usando o teste *post-hoc* de Bonferroni–Dunn.

6.2.2.3 Avaliação do `MDLText` na classificação de documentos de texto curtos e ruidosos

O `MDLText` também foi avaliado em um cenário desafiador para os métodos tradicionais: a classificação de documentos de texto curtos e ruidosos. O objetivo foi analisar se o desempenho é afetado em aplicações que envolvem um baixo número de termos por documento, agravado pela presença de ruídos como gírias, expressões idiomáticas, símbolos, *emoticons* e abreviações. Problemas de classificação de texto com tais características são altamente desafiadores, pois com um baixo número de termos, menos informações estão disponíveis para serem utilizadas pelo modelo de predição. Ainda, os ruídos mencionados dificultam a extração dos atributos e a geração do modelo espaço-vetorial (ALMEIDA *et al.*, 2016).

Alguns problemas que envolvem a classificação de documentos de texto curtos e ruidosos incluem a análise de sentimentos em comentários de redes sociais, detecção de comentários ofensivos, detecção de pedofilia em mensagens instantâneas (IM – *instant messages*) e detecção de spam em serviço de mensagens curtas (SMS - *short message service*), IM, comentários em Blogs e sites de compartilhamento de vídeo e avaliações *online* de produtos.

Para avaliar o `MDLText` neste cenário, foram realizados experimentos com as seguintes bases de dados de documentos de texto curtos publicamente disponíveis:

- *SMS Spam Collection* (ALMEIDA *et al.*, 2011): composta por 5.574 mensagens SMS extraídas de diferentes fontes;
- *Blog Spam Collection* (MISHNE *et al.*, 2005): composta por 1.024 comentários de texto extraídos de 50 *posts* aleatórios de *blogs*. Os comentários foram manualmente

rotulados;

- *Review Spam Collection* (OTT *et al.*, 2011; OTT *et al.*, 2013): composta por 1.600 avaliações sobre os 20 hotéis mais populares da cidade de Chicago;
- *YouTube Spam Collection (YSC)* (ALBERTO *et al.*, 2015): coleção composta por cinco bases de dados, cada uma contendo comentários de texto extraídos dos vídeos mais visualizados do Youtube (Eminem, Katy Perry, LMFAO, Psy e Shakira).

As estatísticas das bases de dados utilizadas nos experimentos são apresentadas na Tabela 14, onde $|\mathcal{D}|$ é a quantidade de documentos de texto de cada base de dados, $|\mathcal{T}|$ corresponde ao número de atributos (tamanho do vocabulário), $|S|$ é o grau de esparsidade da base de dados, ψ é a mediana do número de termos por documento, IQR é a amplitude interquartil do número de termos por documento e $|\mathcal{C}|$ é o número de classes. A última coluna apresenta a quantidade de documentos de cada classe.

Tabela 14 – Bases de dados de documentos de texto curtos e ruidosos usadas nos experimentos.

Base de dados	$\|\mathcal{D}\|$	$\|\mathcal{T}\|$	$\|S\|$	ψ	IQR	$\|\mathcal{C}\|$	Tamanho das classes
Blog	1.024	8.163	99,6%	22.0	35.5	2	332, 692
Review	1.600	9.571	99,0%	84.0	51.0	2	800, 800
SMS	5.574	8.706	99,8%	11.0	13.0	2	747, 4.827
YSC - Eminem	448	1.601	99,0%	7.0	13.5	2	203, 245
YSC - Katy Perry	350	1.755	99,1%	11.0	12.0	2	175, 175
YSC - LMFAO	438	954	99,1%	6.0	5.0	2	202, 236
YSC - Psy	350	1.429	99,1%	9.0	9.0	2	175, 175
YSC - Shakira	370	1.357	98,9%	6.0	12.0	2	174, 196

Como pode ser observado na Tabela 14, a mediana do número de termos por documento é abaixo de 20 na maioria das bases de dados, o que comprova que elas são compostas por documentos de texto curtos. Somado a isso, todas essas bases de dados são compostas por documentos gerados por usuários. Portanto, eles são repletos de ruídos, como abreviações e erros ortográficos. A Tabela 15 apresenta alguns documentos curtos e ruidosos das bases de dados usadas nos experimentos.

Em cenários como esses, é recomendado empregar técnicas de normalização léxica e indexação semântica para minimizar os problemas de representação e, consequentemente, auxiliar os métodos de classificação. Porém, nesta seção, essas técnicas não foram aplicadas, pois o objetivo principal não foi obter os melhores resultados de classificação, mas avaliar o desempenho do `MDLText` neste problema, sem o auxílio de tais técnicas (consulte o Apêndice A para conferir os resultados obtidos após o uso das técnicas de normalização léxica e indexação semântica). A única técnica de pré-processamento aplicada foi a conversão para letras minúsculas. Além disso, na etapa de *tokenização*, os delimitadores utilizados foram quaisquer caracteres não alfanuméricos.

Tabela 15 – Exemplos de documentos de texto curtos e ruidosos.

Ok i msg u b4 i leave my house.
U dun say so early hor... U c already then say...
Are you unique enough? Find out from 30th August. www.areyouunique.co.uk
FREE entry into our $250 weekly competition just text the word WIN to 80086 NOW. 18 T&C www.txttowin.co.uk
You have won ?1,000 cash or a ?2,000 prize! To claim, call09050000327
OMG LISTEN TO THIS ITS SOO GOOD!! :D
subscribe me if u love eminem!
Hey plz check out my music video. Thanks!! :-)
Free itunes $25 giftcard codes: http://shhort.com/a?r=00CnjqU2b
She kinda let herself go, huh?
Plz help me getting 1.000 Subscribers tonight/today. Thanks to all who sub me :)
Plizz withing my channel
thumbs up if u checked this video to see hw views it got
Very useful comments - good to readonline casinos (online casino) choice
check out the real poker online at this cool site.

Na Tabela 16, são mostrados os resultados de cada método avaliado para cada uma das bases de dados apresentadas na Tabela 14, onde os valores correspondem à média da macro F-medida obtida por cada método de classificação em dez rodadas do experimento descrito no Algoritmo 6.1.

Tabela 16 – Média da macro F-medida obtida por cada método na classificação *online* dos documentos de texto curtos e ruidosos em 10 rodadas.

	B.NB	M.NB	MDL	`MDLText`	OGD	Perceptron	ROMMA	SGD
Blog	0,803	**0,858**	0,819	0,841	0,815	0,784	0,762	0,809
Review	**0,843**	0,841	0,770	0,835	0,802	0,789	0,712	0,804
SMS	0,892	**0,955**	0,921	0,949	0,718	0,917	0,902	0,928
YSC - Eminem	**0,913**	0,891	0,887	0,907	0,790	0,874	0,862	0,894
YSC - Katy Perry	**0,911**	0,900	0,867	0,902	0,855	0,865	0,847	0,873
YSC - LMFAO	**0,924**	0,906	0,897	0,899	0,807	0,886	0,856	0,901
YSC - Psy	0,940	**0,941**	0,897	0,939	0,905	0,896	0,862	0,911
YSC - Shakira	0,875	0,909	0,898	**0,911**	0,885	0,896	0,891	0,901

O `MDLText` obteve bons resultados na classificação dos documentos curtos e ruidosos. Nos experimentos com a base de dados `YSC - Shakira` ele obteve desempenho superior a todos os outros métodos. Além disso, ele foi um dos três melhores métodos para todas as outras bases de dados.

Para uma melhor análise dos resultados, a Figura 25 apresenta os *rankings* médios obtidos por todos os métodos avaliados. Conforme pode ser observado, o `MDLText` obteve o segundo melhor *ranking* médio.

Foi realizada uma análise estatística dos resultados usando o teste de Friedman com um intervalo de confiança $\alpha = 0{,}05$. Essa análise indicou que a hipótese nula de igual-

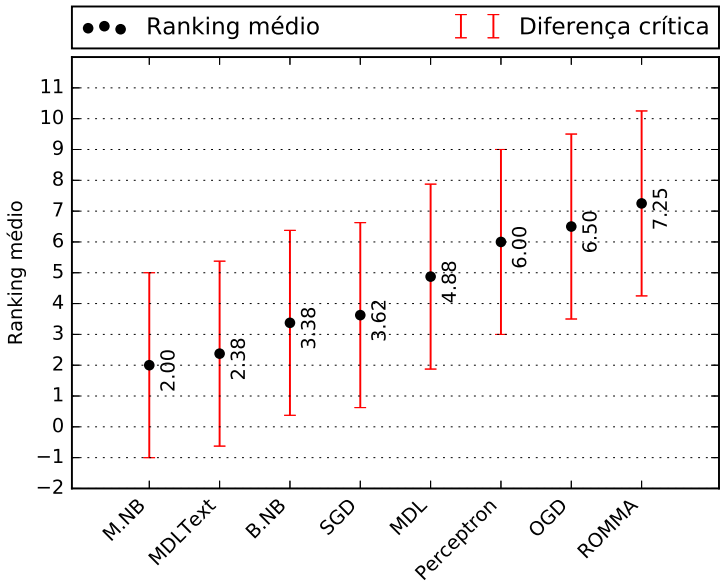


Figura 25 – *Ranking* médio obtido por cada método na classificação de documentos curtos e ruidosos e diferenças críticas calculadas usando o teste *post-hoc* de Bonferroni–Dunn.

dade entre os métodos pode ser descartada. Diante disso, foi feita uma análise estatística par-a-par entre os métodos usando o teste *post-hoc* de Bonferroni–Dunn, com um intervalo de confiança $\alpha = 0{,}05$. Para esse intervalo de confiança, a diferença crítica calculada pelo método foi igual a 3,001. Portanto, a análise estatística indicou que o `MDLText` obteve um desempenho significativamente superior aos métodos Perceptron, OGD e ROMMA. Porém, ele foi considerado estatisticamente equivalente aos demais.

6.3 Considerações finais

Neste capítulo, foram apresentados os experimentos realizados para avaliar o desempenho do `MDLText` em cenários de aprendizado *offline* e *online*.

Os experimentos foram realizados usando 45 bases de dados grandes, reais e públicas e os resultados obtidos pelo `MDLText` foram comparados aos obtidos por métodos de aprendizado tradicionais da literatura. Para assegurar uma comparação justa, foi usada uma busca em grade para encontrar o melhor esquema de pesos e os melhores parâmetros para cada método.

Em ambos cenários de aprendizado (*offline* e *online*) foi verificado que o `MDLText` obteve melhor desempenho quando a técnica CF foi usada para calcular a pontuação dos termos que é usada na função que calcula o tamanho de descrição das classes. Portanto, essa técnica foi usada em todas as comparações feitas entre o `MDLText` e os outros métodos de classificação.

No cenário de aprendizado *offline*, a análise estatística dos resultados mostrou

que o `MDLText` obteve um desempenho melhor que os métodos M.NB, B.NB, KNN, DT, RF e Rocchio. A mesma análise indicou que não houve uma diferença significativa entre o `MDLText` e o SVM. Porém, a análise do tempo computacional entre os dois métodos mostrou que o `MDLText` é muito mais rápido que o SVM em problemas de grande porte.

Também, foi analisado o impacto da seleção de atributos no desempenho dos métodos e verificou-se que a maioria deles, incluindo o `MDLText`, foi pouco afetada.

Posteriormente, foram realizados experimentos em um cenário de aprendizado *online*. A análise estatística dos resultados mostrou que o `MDLText` obteve desempenho significativamente melhor que todos os demais métodos comparados.

O método proposto também foi avaliado em um problema de classificação de textos curtos e ruidosos. Assim como nos outros cenários, ele apresentou bom desempenho, pois seu *ranking* médio foi o segundo melhor. Portanto, mesmo com a grande presença de ruídos e com poucas informações para serem extraídas, devido à baixa quantidade de termos por documento, a capacidade de predição do `MDLText` foi pouco afetada, se comparada aos outros métodos avaliados.

De forma geral, o `MDLText` mostrou ser eficiente e robusto, mesmo em cenários desafiadores. Ele obteve resultados melhores que a maioria dos métodos avaliados tanto em problemas com alta dimensionalidade, quanto em problemas em que a quantidade de termos por documento é baixa. Além disso, ele apresentou bom desempenho nas bases de dados altamente desbalanceadas. A quantidade de classes também não afetou o desempenho geral do método proposto. Por fim, o `MDLText` apresentou baixo custo computacional e alto poder de predição, associado à flexibilidade de ser usado tanto em cenários de classificação *offline*, quanto *online*.

7 O método MDLClass

Neste capítulo, é apresentada uma proposta inicial de um método de classificação baseado no princípio MDL. Trata-se de uma generalização do MDLText para lidar com problemas de classificação não-textuais, cujas instâncias possam ser representadas por atributos contínuos ou categóricos.

Primeiramente, é apresentada a base matemática do método proposto. Depois, é apresentado o estudo da sua complexidade computacional. O desempenho do MDLClass é comparado aos resultados de métodos tradicionais de classificação *offline*, usando 23 bases de dados públicas e bem conhecidas, compostas por atributos numéricos ou categóricos.

7.1 Base matemática

Cada exemplo de treinamento d em uma base de dados $\mathcal{D}$ pode ser definido como um conjunto de $|\mathcal{B}|$ atributos categóricos $\mathcal{B} = \{b_1, b_2, ..., b_{|\mathcal{B}|}\}$. Cada atributo b_i possui $|b_i|$ valores distintos $\{v_1(b_i), v_2(b_i), ..., v_{|b_i|}(b_i)\}$. No método proposto, cada possível valor de um atributo b_i é transformado em um atributo binário. Esse processo é chamado de codificação de um *bit* por estado (OHE – *one-hot encoding*) e gera uma quantidade de atributos binários igual ao número de valores distintos do atributo original. O atributo binário correspondente ao valor do atributo atual, recebe o valor um, enquanto os demais recebem o valor zero. Portanto, cada exemplo d pode ser definido por um novo conjunto de atributos $\mathcal{T} = \{b_1 = v_1(b_1), ..., b_1 = v_{|b_1|}(b_1), b_2 = v_1(b_2), ..., b_2 = v_{|b_2|}(b_2), ..., b_{|\mathcal{B}|} = v_1(b_{|\mathcal{B}|}), ..., b_{|\mathcal{B}|} = v_{|b_{|\mathcal{B}|}|}(b_{|\mathcal{B}|})\}$. Por simplicidade, o novo conjunto de atributos será representado por $\mathcal{T} = \{t_1, t_2, ..., t_{|\mathcal{T}|}\}$.

Se os valores do atributo são numéricos, antes de aplicar a codificação OHE, eles são discretizados por meio do método proposto por Fayyad e Irani (1993) que é amplamente empregado na literatura.

Assim como é feito no MDLText, dado um exemplo d não-rotulado, o MDLClass utiliza a equação principal do princípio MDL (Equação 4.9) para predizer a classe à qual o exemplo não-rotulado pertence. Portanto, d recebe o rótulo j correspondente à classe c_j que possui o menor tamanho de descrição em relação à d:

$$c(d) = \arg\min_{\forall c} L(d|c_j). \tag{7.1}$$

O tamanho da descrição $L(d|c_j)$ corresponde à soma do tamanho da descrição de todos os atributos de d maiores que zero, gerados após aplicar OHE, multiplicado por uma penalidade $K(t_i)$ atribuída a cada um deles:

$$L(d|c_j) = \sum_{\forall i|t_i>0} L(t_i|c_j) \times K(t_i), \tag{7.2}$$

onde $L(t_i|c_j) = \lceil -\log_2 \beta(t_i|c_j) \rceil$. O termo $\beta(t_i|c_j)$ corresponde à probabilidade condicional do atributo t_i dada a classe c_j, que pode ser calculada da seguinte maneira:

$$\beta(t_i|c_j) = \frac{n_{c_j,t_i} + \frac{1}{|\Omega|}}{n_{c_j} + 1}, \tag{7.3}$$

onde n_{c_j,t_i} corresponde ao número de exemplos de treinamento da classe c_j em que $t_i > 0$ e n_{c_j} é a soma de n_{c_j,t_i} para todos os atributos dos exemplos de treinamento pertencentes à classe c_j. O parâmetro $|\Omega|$ é usado para preservar uma porção do tamanho de descrição para os atributos em que $n_{c_j,t_i} = 0$. Ele também regula o quanto tais atributos irão contribuir para o tamanho de descrição do exemplo dada a classe c_j. A influência exercida por esse parâmetro no valor de $L(t_i|c_j)$ é semelhante ao `MDLText`, conforme ilustrado pela Figura 6.

A função de penalidade $K(t_i)$ é calculada pela seguinte equação:

$$K(t_i) = \log\left(\frac{1}{(1+\mu) - F(t_i)}\right)^e + 1, \tag{7.4}$$

onde $0 \leq F(t_i) \leq 1$ é uma pontuação dada ao atributo t_i, $e \geq 1$ é um parâmetro que ajusta a velocidade de crescimento de $K(t_i)$ e $\mu > 0$ é uma constante usada para evitar que o denominador seja zero. O valor usado nesta tese foi $\mu = 10^{-3}$.

Como pode ser observado, a equação que calcula o valor de $K(t_i)$ no `MDLClass` possui um parâmetro a mais do que foi usado no `MDLText`. Esse novo parâmetro ($e \geq 1$) tem a finalidade de tornar o método mais flexível e ajustar a penalidade para problemas não textuais. A Figura 26 mostra como a função de penalidade $K(t_i)$ varia de acordo com $F(t_i)$ para diferentes valores de e.

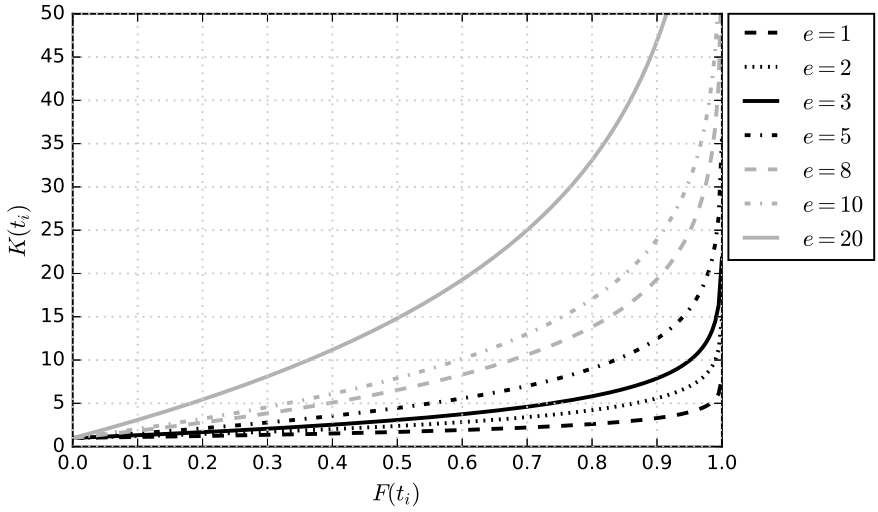


Figura 26 – Variação de $K(t_i)$ de acordo com $F(t_i)$ para diferentes valores de e.

A função de penalidade $K(t_i)$ usada no `MDLClass` tem a finalidade de aumentar a capacidade de separação das classes e, assim como no `MDLText`, ela retorna valores maiores para atributos mais discriminativos. Quando a frequência do atributo é semelhante

para todas as classes, ele tende a receber uma pontuação baixa. Quando a frequência do atributo é alta em relação a uma classe e baixa em relação às outras, sua pontuação tende a ser alta.

Para calcular $F(t_i)$ pode ser usada qualquer função que retorne uma pontuação $0 \leq F(t_i) \leq 1$. Portanto, as técnicas apresentadas na Seção 5.1.2 também podem ser aplicadas para calcular a pontuação dos atributos no `MDLClass`.

Os algoritmos de treinamento e predição do `MDLClass` são semelhantes aos Algoritmos 5.1 e 5.2 referentes ao `MDLText` e, portanto, dispensam apresentação.

7.1.1 Análise da complexidade computacional

No estágio de treinamento, cada atributo de cada exemplo de treinamento necessita ser visitado uma única vez. Portanto, dados $|\mathcal{D}^\star|$ exemplos de treinamento e $|\mathcal{B}|$ atributos, a complexidade computacional desse estágio é de ordem linear $\mathcal{O}(|\mathcal{D}^\star| \times |\mathcal{B}|)$.

No estágio de predição, para cada exemplo que precisa ser classificado, é necessário calcular a pontuação $K(t_i)$ de cada atributo que, posteriormente, é usada para calcular o tamanho de descrição relativo a cada classe. Portanto, a complexidade computacional é $\mathcal{O}(|\mathcal{C}| \times |\mathcal{B}| + |\mathcal{C}| \times |\mathcal{B}|)$ ou $\mathcal{O}(2 \times |\mathcal{C}| \times |\mathcal{B}|)$, onde $|\mathcal{C}|$ é o número de classes. Como $2 \times |\mathcal{C}|$ é constante, a complexidade computacional do estágio de predição do `MDLClass` é de ordem linear $\mathcal{O}(|\mathcal{B}|)$.

É importante notar que a esparsidade inversa dos vetores de atributos gerados após aplicar OHE é igual ao número original de atributos ($|\mathcal{B}|$). Portanto, na análise da complexidade do `MDLClass`, foi considerado o número original de atributos do problema.

7.2 Metodologia

Para avaliar o desempenho do `MDLClass`, foram realizados experimentos com 23 bases de dados públicas e bem conhecidas, cujos exemplos são representados por atributos categóricos ou numéricos. As estatísticas dessas bases são apresentadas na Tabela 17, onde $|\mathcal{D}|$ é a quantidade de exemplos de cada base de dados, $|\mathcal{B}|$ corresponde ao número de atributos e $|\mathcal{C}|$ é o número de classes. A última coluna apresenta a quantidade de exemplos em cada classe.

As bases de dados WEBSPAM-UK2006 e WEBSPAM-UK2007[1] são formadas por páginas Web de diversos *hosts*. Elas foram disponibilizadas por meio de três conjuntos de atributos: o primeiro é composto por 96 atributos baseados no conteúdo das páginas Web (ABC) (CASTILLO *et al.*, 2007), o segundo é composto por 41 atributos baseados nos *links* das páginas Web (ABL) (BECCHETTI *et al.*, 2006) e o terceiro é composto

[1] Yahoo! Research: "Web Spam Collections". Disponível em ⟨http://chato.cl/webspam/⟩. Acessado em: 19/01/2017.

Tabela 17 – Bases de dados usadas nos experimentos.

Base de dados	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{B}\rvert$	$\lvert\mathcal{C}\rvert$	Tamanho das classes
Balance-scale	625	4	3	288, 288, 49
Binary alphadigits	1.404	320	36	39 (cada classe)
Breast cancer Wisconsin	699	9	2	458, 241
Car evaluation	1.728	6	4	1.210, 384, 69, 65
King+Rook vs King+Pawn	3.196	36	2	1.669, 1527
Monks	556	6	2	278 (cada classe)
SPECT	267	22	2	212, 55
Tic-tac-toe	958	9	2	626, 332
UK2006-ABC	8.487	96	2	6.509, 1.978
UK2006-ABL	8.487	41	2	6.509, 1.978
UK2006-ABLT	8.487	138	2	6.509, 1.978
UK2006-ABL+ABC	8.487	137	2	6.509, 1.978
UK2006-ABL+ABLT	8.487	179	2	6.509, 1.978
UK2006-ABLT+ABC	8.487	234	2	6.509, 1.978
UK006-ABL+ABLT+ABC	8.487	275	2	6.509, 1.978
UK2007-ABC	5.797	96	2	5.476, 321
UK2007-ABL	5.797	41	2	5.476, 321
UK2007-ABLT	5.797	138	2	5.476, 321
UK2007-ABL+ABC	5.797	137	2	5.476, 321
UK2007-ABL+ABLT	5.797	179	2	5.476, 321
UK2007-ABLT+ABC	5.797	234	2	5.476, 321
UK2007-ABL+ABLT+ABC	5.797	275	2	5.476, 321
Voting records	435	16	2	267, 168

por 138 atributos baseados nos *links* transformados (ABLT) (CASTILLO *et al.*, 2007). Também, foram feitos experimentos com todas as possíveis combinações entre os três tipos de atributos (ABC, ABL e ABLT). Todos os exemplos que possuem uma classe indefinida ou que não são rotulados foram removidos.

As outras bases de dados usadas nos experimentos e apresentadas na Tabela 17 foram extraídas do repositório UCI[2] que é amplamente conhecido pela comunidade de aprendizado de máquina.

7.3 Avaliação

Os resultados obtidos pelo `MDLClass` foram comparados aos resultados de outros métodos amplamente usados como *baseline* em trabalhos de classificação: M.NB (MCCALLUM; NIGAM, 1998), B.NB (MCCALLUM; NIGAM, 1998), SVM (BOSER *et al.*, 1992; CORTES; VAPNIK, 1995), KNN (COVER; HART, 1967), DT (CART) (BREIMAN

[2] O repositório UCI (UC Irvine *Machine Learning Repository*) está disponível em ⟨http://archive.ics.uci.edu/ml/⟩. Acessado em: 19/01/2017.

et al., 1984a) e RF (BREIMAN, 2001).

Os métodos M.NB, B.NB, KNN, DT e RF foram implementandos em `Python` usando a biblioteca `scikit-learn Library`. O `MDLClass` foi implementado em `MATLAB`. Além disso, em todos os experimentos com o `MDLClass`, a técnica CF foi usada para calcular a pontuação dos atributos e o parâmetro $|\Omega|$ foi usado com o valor de 2^{30}. O método SVM foi implementado usando a biblioteca `LibSVM` para `C++`. O *kernel* utilizado nos experimentos com o SVM foi o RBF.

O desempenho de alguns métodos de classificação em problemas que possuem atributos numéricos pode ser melhorado com a discretização dos valores dos atributos. Portanto, foram realizados experimentos com (1) os atributos normalizados pela normalização Z-*score* e com (2) os atributos discretizados pelo método de Fayyad e Irani (1993).

O desempenho dos métodos KNN, RF, SVM e `MDLClass` pode ser bastante afetado pela escolha dos seus parâmetros. Portanto, foi realizada uma busca em grade usando a validação cruzada 5-*fold* estratificada para encontrar o melhor valor para o número de vizinhos (KNN), o número de árvores (RF), o parâmetro e (`MDLClass`) e os parâmetros custo e γ (SVM).

7.4 Resultados

A Tabela 18 apresenta os resultados obtidos por cada método avaliado para cada uma das 23 bases de dados. Os resultados foram calculados usando a macro F-medida e validação cruzada 5-*fold* estratificada. Os valores em negrito indicam o melhor resultado.

Os resultados indicam que o `MDLClass` obteve resultados melhores que os métodos M.NB, B.NB e DT na maioria dos experimentos. Porém, em geral, ele obteve resultados inferiores aos métodos RF, SVM e KNN. Apesar disso, o método proposto tem como vantagem o fato de ter complexidade computacional linear. Portanto, o `MDLClass` pode ser uma boa alternativa em problemas que demandam respostas rápidas.

Para uma análise mais confiável dos resultados foi feita uma análise estatística considerando os *rankings* médios apresentados na Figura 27.

Primeiramente, foi usado o teste não-paramétrico de Friedman para verificar se existe diferença estatística entre os métodos avaliados. Para 7 métodos e 23 bases de dados, usando um intervalo de confiança $\alpha = 0{,}05$, o teste de Friedman indicou que a hipótese nula de que todos os métodos são equivalentes pode ser rejeitada. Em seguida, foi feita uma análise par-a-par usando o teste *post-hoc* de Bonferroni–Dun. Para um intervalo de confiança $\alpha = 0{,}05$, a diferença crítica foi igual a 1,681. Portanto, não há uma diferença significativa entre o desempenho do `MDLClass` e o desempenho dos métodos KNN, M.NB, B.NB e DT. Porém, há evidência estatística suficiente para afirmar que o método proposto foi inferior aos métodos SVM e RF.

Tabela 18 – Macro F-medida obtida por cada método usando validação cruzada 5-*fold* estratificada.

	B.NB	DT	KNN	M.NB	`MDLClass`	RF	SVM
Balance-scale	0,634	0,554	0,593	0,633	0,622	0,585	**0,951**
Binary Alphadigits	0,682	0,477	0,709	0,664	0,617	0,719	**0,775**
Breast cancer Wisconsin	0,970	0,922	0,945	**0,972**	0,968	0,959	0,968
Car evaluation	0,741	0,949	0,774	0,669	0,737	0,904	**0,989**
King+Rook vs King+Pawn	0,874	**0,995**	0,956	0,879	0,884	0,991	0,994
Monks	0,787	0,973	0,956	0,787	0,775	**1,000**	**1,000**
SPECT	0,738	0,663	0,712	0,435	**0,778**	0,717	0,691
Tic-tac-toe	0,652	0,923	0,971	0,648	0,670	0,988	**0,994**
UK2006-ABC	0,752	0,769	0,825	0,759	0,772	**0,848**	**0,848**
UK2006-ABL	0,796	0,797	0,828	0,803	0,810	**0,859**	0,849
UK2006-ABLT	0,789	0,799	0,847	0,796	0,810	**0,857**	0,851
UK2006-ABL+ABC	0,832	0,821	0,865	0,830	0,849	**0,887**	0,886
UK2006-ABL+ABLT	0,801	0,801	0,842	0,805	0,821	0,856	**0,861**
UK2006-ABLT+ABC	0,830	0,820	0,867	0,832	0,844	0,885	**0,887**
UK2006-ABL+ABLT+ABC	0,836	0,817	0,864	0,837	0,848	**0,889**	0,888
UK2007-ABC	0,632	0,647	0,689	0,631	0,625	0,732	**0,736**
UK2007-ABL	0,584	0,552	0,555	0,560	0,577	**0,610**	0,559
UK2007-ABLT	0,583	0,545	0,534	**0,590**	0,562	0,545	0,530
UK2007-ABL+ABC	0,639	0,619	0,678	0,635	0,627	0,727	**0,730**
UK2007-ABL+ABLT	0,587	0,536	0,533	**0,591**	0,569	0,550	0,570
UK2007-ABLT+ABC	0,633	0,626	0,674	0,641	0,624	**0,722**	0,676
UK2007-ABL+ABLT+ABC	0,620	0,622	0,637	0,639	0,615	**0,724**	0,673
Voting records	0,899	0,938	0,915	0,902	0,915	**0,961**	0,953

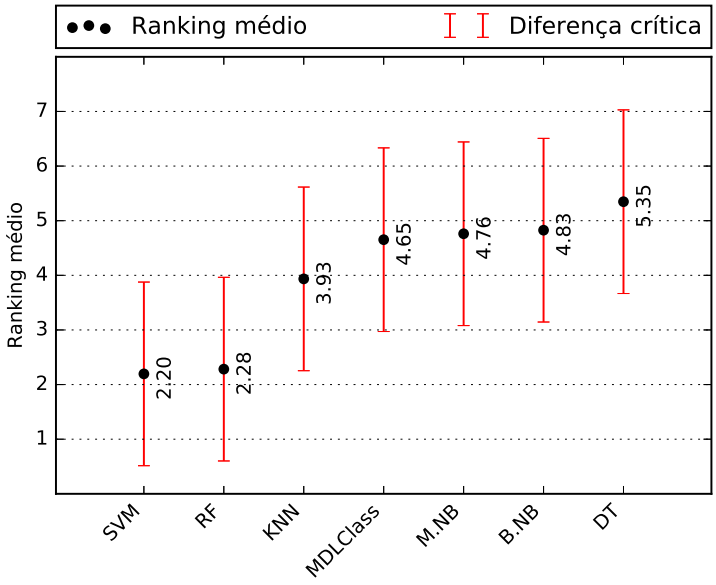


Figura 27 – *Rankings* médios usados na análise estatística e diferenças críticas calculadas usando o teste *post-hoc* de Bonferroni–Dunn.

7.5 Considerações finais

Neste capítulo, foi apresentada uma proposta de generalização do `MDLText`, chamada `MDLClass`. Ele é capaz de classificar instâncias não textuais, compostas por

atributos contínuos ou categóricos.

O `MDLClass` foi comparado a alguns métodos amplamente utilizados como *baseline* em problemas de classificação. A análise estatística dos resultados mostrou que o método proposto foi estatisticamente equivalente aos métodos M.NB, B.NB e DT, apesar de ter obtido um *ranking* médio melhor. Contudo, ele foi estatisticamente inferior aos métodos RF e SVM.

Como o `MDLClass` ainda está em fase inicial e é uma simples proposta de generalização direta do `MDLText`, esses resultados são animadores, pois foram comparáveis aos obtidos por métodos tradicionais da literatura. Além disso, o método proposto tem vantagens sobre alguns métodos clássicos, como o SVM, pois ele pode ser aplicado em problemas multiclasse sem decompô-los em problemas binários. Algumas melhorias precisam ser feitas para que resultados ainda mais expressivos possam ser obtidos. Por exemplo, não foi possível deduzir uma boa função de similaridade entre o documento a ser classificado e as possíveis classes do problema. Como o `MDLClass` usa um vetor de atributos com valores binários, os protótipos das classes obtidos pelo `MDLText` não ajudaram a melhorar a predição das classes no novo método. Portanto, podem ser encontradas novas alternativas para a geração de protótipos para as classes que sejam ao mesmo tempo rápidas e eficientes.

Outro problema em aberto é que, diferentemente do `MDLText`, atualmente o `MDLClass` só pode ser aplicado em problemas de classificação `offline`. O principal motivo para isso é que em problemas que possuem atributos numéricos, ele faz um processo de discretização dos valores. Ainda é desafiador encontrar uma forma de alterar os intervalos de discretização de maneira *online* e ao mesmo tempo atualizar a frequência de vezes em que cada atributo obteve um valor maior que zero para cada classe do problema. Uma alternativa seria usar uma técnica de discretização mais simples, como por exemplo, dividindo-se o intervalo de valores em k partições iguais. Porém, técnicas de discretização mais simples podem diminuir o desempenho da classificação. Outra possibilidade seria não aplicar a discretização, mas empregar diretamente aproximações de funções de distribuição de probabilidade dos atributos através de estimativa de densidade de *kernel*.

8 Conclusões

Nesta tese, foi apresentado um novo método de classificação multinomial de textos baseado no princípio MDL, o `MDLText`. O método proposto tem características desejáveis, tais como simples implementação, aprendizado incremental, baixo custo computacional e robustez na prevenção do problema de sobreajustamento. Ele possui alto poder preditivo, ao mesmo tempo que é eficiente, o que possibilita que ele seja aplicado em problemas de categorização de texto reais, *online* e de grande escala.

Para avaliar o desempenho do método proposto, foram conduzidos experimentos com 45 bases de dados reais, públicas e de grande porte, considerando dois diferentes cenários: aprendizado *offline* e *online*. Os resultados obtidos foram comparados aos de métodos considerados estado-da-arte em problemas de classificação. Para assegurar uma comparação justa, foram realizadas buscas em grade exaustivas para determinar o melhor esquema de pesos para os termos dos documentos e os melhores parâmetros para cada método quando aplicado a cada uma das bases de dados usadas nos experimentos. Dessa maneira, foi garantido que o desempenho do método apresentado nesta tese foi comparado ao melhor desempenho que poderia ser obtido pelos demais métodos.

Os resultados indicaram que o `MDLText` oferece um excelente balanceamento entre poder preditivo e eficiência computacional. A análise estatística dos resultados mostrou que ele foi superior aos métodos M.NB, B.NB, KNN, DT, RF e Rocchio e equivalente ao SVM. Contudo, o MDLText possui aprendizado muito mais eficiente em termos de velocidade de processamento (em média, foi 56 vezes mais rápido do que o método SVM).

Nos experimentos conduzidos em cenário de aprendizado *online*, o `MDLText` foi comparado a métodos de classificação incrementais tradicionais: M.NB, B.NB, Perceptron, SGD, ROMMA e OGD. A análise estatística dos resultados mostrou que o método proposto obteve desempenho superior a todos os outros métodos. Um bom desempenho também foi obtido em um cenário de classificação de textos curtos e ruidosos, onde ele obteve o segundo melhor *ranking* médio.

O `MDLText` pode ser considerado simples, pois seu modelo de predição é baseado apenas em informações do número de ocorrências de cada termo e no peso TF-IDF deles. Ele também pode ser considerado robusto, pois obteve desempenho superior à maioria dos métodos tradicionais de classificação avaliados nesta tese em uma variedade de cenários (classificação *online* e *offline*, bases de dados com classes desbalanceadas, problemas com alta dimensionalidade, problemas binários e multiclasses e textos curtos e ruidosos). Esses bons resultados, aliados ao baixo custo computacional do `MDLText`, o que é atestado pelo fato de que ele possui complexidade linear e foi em média 56 vezes mais rápido que o SVM, mostraram que ele também é eficiente.

Nesta tese, também foi apresentada uma proposta inicial de generalização do `MDLText`, capaz de ser aplicada em problemas não textuais. Esse novo método, nomeado `MDLClass`, pode ser aplicado naturalmente em problemas multiclasse e possui complexidade linear. Para avaliar seu desempenho, foram conduzidos experimentos com 23 bases de dados reais, grandes, públicas e que possuem atributos categóricos ou numéricos. Os resultados obtidos foram comparados ao desempenho de métodos estado-da-arte de classificação. A análise estatística mostrou que o `MDLClass` é estatisticamente equivalente aos métodos KNN, M.NB, B.NB e DT. Porém, ele obteve um desempenho inferior aos métodos SVM e RF. Apesar dos resultados obtidos pelo `MDLClass` terem sido comparáveis aos métodos tradicionais da literatura, por tratar-se de uma proposta inicial, ele ainda é passível de muitas melhorias, que podem aprimorar ainda mais o seu desempenho. Uma função de similaridade, análoga a que foi usada no `MDLText`, adaptada para problemas não-textuais, pode tornar o `MDLClass` mais robusto. Além disso, alterações podem ser feitas para tornar seu aprendizado incremental, como por exemplo: (1) criar um método de discretização mais eficaz e incremental ou (2) calcular a distribuição de probabilidade dos atributos por meio de estimativa de densidade de *kernel*.

8.1 Trabalhos futuros

Durante a elaboração e após a conclusão deste estudo, foram verificados diversos pontos em que este trabalho pode ser melhorado e complementado. A seguir, são apresentados direcionamentos para trabalhos futuros:

- adaptação do `MDLText` para problemas de categorização de texto *multilabel*;
- proposição e avaliação de novas técnicas de atribuição de pontuação para os atributos que superem a técnica CF em problemas de categorização de texto gerais ou problemas com características particulares;
- aplicação do `MDLText` em problemas de categorização de texto em domínios específicos, como por exemplo análise de sentimento, detecção de plágio, detecção de pedofilia em redes sociais, detecção de comentários racistas, entre outros;
- adaptação do `MDLClass` para problemas de classificação *online*;
- proposição de técnica para a geração de protótipos para as classes, para que possam ser usados em uma função de penalidade baseada em similaridade, com o objetivo de melhorar a predição do `MDLClass`;
- estudo de técnicas de discretização mais eficientes e incrementais para serem acopladas ao `MDLClass`; e

- aplicação do `MDLClass` em outros problemas de classificação, como por exemplo previsão de resultados de jogos, detecção de fraudes em transações pela internet, detecção de corrupção de agentes públicos, entre outros.

8.2 Publicações

A seguir são listados os trabalhos provenientes dos resultados de pesquisa reportados nesta tese.

- SILVA, R. M.; ALMEIDA, T. A.; YAMAKAMI, A. `MDLText`: An Efficient and Lightweight Text Classifier. *Knowledge-Based Systems*, Elsevier, 118:152–164, 2017. doi: ⟨http://dx.doi.org/10.1016/j.knosys.2016.11.018⟩.

- SILVA, R. M.; ALMEIDA, T. A.; YAMAKAMI, A. Towards filtering undesired short text messages using an online learning approach with semantic indexing. *Expert Systems With Applications*, Elsevier. ***Em revisão**.

- SILVA, R. M.; ALMEIDA, T. A.; YAMAKAMI, A. Towards Web Spam Filtering using a Classifier based on the Minimum Description Length Principle. *Proceedings of the 15th IEEE International Conference on Machine Learning and Applications (ICMLA'16)*. Anaheim, California, USA: IEEE, p. 1–6, 2016.

- SILVA, R. M.; ALMEIDA, T. A.; YAMAKAMI, A. Detecção Automática de SPIM e SMS Spam usando Método baseado no Princípio da Descrição mais Simples. *Anais do 13th Encontro Nacional de Inteligência Artificial e Computacional (ENIAC'16)*. Recife, PE, Brasil, 2016. p. 181–192.

- SILVA, R. M.; ALBERTO, T. A; ALMEIDA, T. A.; YAMAKAMI, A. Filtrando Comentários do YouTube através de Classificação Online baseada no Princípio MDL e Indexação Semântica. *Anais do 13th Encontro Nacional de Inteligência Artificial e Computacional (ENIAC'16)*. Recife, PE, Brasil, 2016. p. 193–204.

- SILVA, R. M.; ALMEIDA, T. A.; YAMAKAMI, A. Quanto mais simples, melhor! Categorização de Textos baseada na Navalha de Occam. *Anais do 12th Encontro Nacional de Inteligência Artificial e Computacional (ENIAC'15)*. Natal, RN, Brasil, 2015. p. 1–7.

Referências

AGGARWAL, C. C. *Data Classification: Algorithms and Applications*. 1st. ed. [S.l.]: Chapman & Hall/CRC, 2014. ISBN 1466586745, 9781466586741. Citado 2 vezes nas páginas 31 e 42.

AHMED, I.; ALI, R.; GUAN, D.; LEE, Y.-K.; LEE, S.; CHUNG, T. Semi-supervised learning using frequent itemset and ensemble learning for SMS classification. *Expert Systems with Applications*, Pergamon Press, Inc., Tarrytown, NY, USA, v. 42, n. 3, p. 1065–1073, fev. 2015. Citado na página 37.

ALBERTO, T. C.; LOCHTER, J. V.; ALMEIDA, T. A. Tubespam: Comment spam filtering on youtube. In: *Proceedings of the 14th International Conference on Machine Learning and Applications (ICMLA'15)*. Miami, FL, USA: IEEE, 2015. p. 138–143. Citado 2 vezes nas páginas 37 e 93.

ALMEIDA, T. A.; HIDALGO, J. M. G.; YAMAKAMI, A. Contributions to the study of SMS spam filtering: new collection and results. In: *Proceedings of the 11th ACM Symposium on Document engineering (DocEng'11)*. Mountain View, CA, USA: ACM, 2011. p. 259–262. Citado na página 92.

ALMEIDA, T. A.; SILVA, T. P.; SANTOS, I.; HIDALGO, J. M. G. Text normalization and semantic indexing to enhance instant messaging and SMS spam filtering. *Knowledge-Based Systems*, Elsevier, v. 108, p. 25–32, maio 2016. Citado 10 vezes nas páginas 11, 37, 41, 73, 92, 120, 121, 122, 127 e 129.

ALMEIDA, T. A.; YAMAKAMI, A. Facing the spammers: A very effective approach to avoid junk e-mails. *Expert Systems with Applications*, Elsevier, v. 39, n. 7, p. 6557–6561, jun. 2012. Citado 3 vezes nas páginas 23, 56 e 58.

ALMEIDA, T. A.; YAMAKAMI, A.; ALMEIDA, J. Filtering spams using the minimum description length principle. In: *Proceedings of the 2010 ACM Symposium on Applied Computing (SAC'10)*. Sierre, Switzerland: ACM, 2010. p. 1854–1858. Citado 3 vezes nas páginas 23, 56 e 58.

ALMEIDA, T. A.; YAMAKAMI, A.; ALMEIDA, J. Spam filtering: how the dimensionality reduction affects the accuracy of naive Bayes classifiers. *Journal of Internet Services and Applications*, Springer-Verlag, v. 1, n. 3, p. 183–200, fev. 2011. Citado 5 vezes nas páginas 23, 33, 35, 37 e 73.

ALSALEH, M.; ALARIFI, A.; AL-QUAYED, F.; AL-SALMAN, A. Combating comment spam with machine learning approaches. In: *Proceedings of the 14th International Conference on Machine Learning and Applications (ICMLA'15)*. Miami, FL, USA: IEEE, 2015. p. 295–300. Citado na página 37.

APHINYANAPHONGS, Y.; FU, L. D.; LI, Z.; PESKIN, E. R.; EFSTATHIADIS, E.; ALIFERIS, C. F.; STATNIKOV, A. A comprehensive empirical comparison of modern supervised classification and feature selection methods for text categorization. *Journal of the Association for Information Science and Technology*, Wiley Online Library, v. 65, n. 10, p. 1964–1987, 2014. ISSN 2330-1643. Citado 2 vezes nas páginas 37 e 73.

ASSIS, F.; YERAZUNIS, W.; SIEFKES, C.; CHHABRA, S. Exponential differential document count – a feature selection factor for improving Bayesian filters accuracy. In: *Proceedings of the 2006 MIT Spam Conference (SP'06)*. Cambridge, MA, USA: [s.n.], 2006. p. 1–6. Citado 2 vezes nas páginas 24 e 65.

BAGNELL, J. A.; BRADLEY, D.; SILVER, D.; SOFMAN, B.; STENTZ, A. Learning for autonomous navigation. *IEEE Robotics Automation Magazine*, IEEE, v. 17, n. 2, p. 74–84, jun. 2010. ISSN 1070-9932. Citado na página 27.

BAHNSEN, A. C.; AOUADA, D.; STOJANOVIC, A.; OTTERSTEN, B. Feature engineering strategies for credit card fraud detection. *Expert Systems with Applications*, Elsevier, v. 51, p. 134–142, 2016. Citado na página 27.

BARRON, A.; RISSANEN, J.; YU, B. The minimum description length principle in coding and modeling. *IEEE Transaction on Information Theory*, v. 44, n. 6, p. 2743–2760, out. 1998. Citado 2 vezes nas páginas 52 e 53.

BAXTER, R. A.; OLIVER, J. *MDL and MML: Similarities and Differences (Introduction to Minimum Encoding Inference – Part III)*. Department of Computer Science, Monash University, Clayton, Victoria, Australia, 1994. Citado na página 52.

BECCHETTI, L.; CASTILLO, C.; DONATO, D.; LEONARDI, S.; BAEZA-YATES, R. Using rank propagation and probabilistic counting for link-based spam detection. In: *Proceedings of the 2006 Workshop on Web Mining and Web Usage Analysis (WebKDD'06)*. Philadelphia,USA: [s.n.], 2006. Citado na página 99.

BEGUM, N.; HU, B.; RAKTHANMANON, T.; KEOGH, E. Towards a minimum description length based stopping criterion for semi-supervised time series classification. In: *Proceedings of the 14th IEEE International Conference on Information Reuse and Integration (IRI'13)*. San Francisco, CA, USA: IEEE, 2013. p. 333–340. Citado na página 55.

BEN-DAVID, A. Comparison of classification accuracy using Cohen's weighted kappa. *Expert Systems with Applications*, Elsevier, v. 34, n. 2, p. 825–832, fev. 2008. Citado na página 35.

BISHOP, C. M. *Pattern Recognition and Machine Learning*. 1th. ed. New York, NY, USA: Springer, 2006. Citado 3 vezes nas páginas 28, 29 e 31.

BOLÓN-CANEDO, V.; SÁNCHEZ-MARROÑO, N.; ALONSO-BETANZOS, A. Recent advances and emerging challenges of feature selection in the context of big data. *Knowledge-Based Systems*, Elsevier, v. 86, p. 33–45, 2015. ISSN 0950-7051. Citado 2 vezes nas páginas 44 e 45.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory (COLT'92)*. Pittsburgh, PA, USA: ACM, 1992. p. 144–152. Citado 2 vezes nas páginas 73 e 100.

BOSIN, A.; DESSÌ, N.; PES, B. High-dimensional micro-array data classification using minimum description length and domain expert knowledge. In: *Advances in Applied Artificial Intelligence*. [S.l.]: Springer Berlin Heidelberg, 2006, (Lecture Notes in Computer Science, v. 4031). p. 790–799. Citado na página 54.

BOUCHARD, G.; CELEUX, G. Selection of generative models in classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, v. 28, n. 4, p. 544–554, abr. 2006. ISSN 0162-8828. Citado na página 47.

BRBIĆ, M.; ŽARKO, I. P. Tuning machine learning algorithms for content-based movie recommendation. *Intelligent Decision Technologies*, v. 9, n. 3, p. 233–242, 2015. Citado na página 27.

BREIMAN, L. Bagging predictors. *Machine Learning*, Kluwer Academic Publishers, Hingham, MA, USA, v. 24, n. 2, p. 123–140, ago. 1996. Citado na página 30.

BREIMAN, L. Random forests. *Machine Learning*, Kluwer Academic Publishers, Hingham, MA, USA, v. 45, n. 1, p. 5–32, out. 2001. Citado 3 vezes nas páginas 30, 73 e 101.

BREIMAN, L.; FRIEDMAN, J.; STONE, C. J.; OLSHEN, R. A. *Classification and regression trees*. [S.l.]: CRC press, 1984. Citado 2 vezes nas páginas 73 e 101.

BREIMAN, L.; FRIEDMAN, J. H.; OLSHEN, R. A.; STONE, C. J. *Classification and Regression Trees*. Belmont, California, USA: Wadsworth International Group, 1984. Citado 2 vezes nas páginas 23 e 30.

CASTILLO, C.; DONATO, D.; GIONIS, A. Know your neighbors: Web spam detection using the web topology. In: *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'07)*. Amsterdam, The Netherlands: [s.n.], 2007. p. 423–430. Citado 2 vezes nas páginas 99 e 100.

CHAITANKAR, V.; ZHANG, C.; GHOSH, P.; PERKINS, E. J.; GONG, P.; DENG, Y. Gene regulatory network inference using predictive minimum description length principle and conditional mutual information. In: *Proceedings of the 2009 International Joint Conference on Bioinformatics, Systems Biology and Intelligent Computing (IJCBS'09)*. Shanghai, China: IEEE, 2009. p. 333–340. Citado na página 54.

CHAITIN, G. J. On the length of programs for computing finite binary sequences: statistical considerations. *Journal of the Association for Computing Machinery (JACM)*, ACM, v. 16, n. 1, p. 145–159, jan. 1969. Citado na página 52.

CHAN, P. P.; YANG, C.; YEUNG, D. S.; NG, W. W. Spam filtering for short messages in adversarial environment. *Neurocomputing*, Elsevier, Amsterdam, The Netherlands, The Netherlands, v. 155, n. C, p. 167–176, maio 2015. Citado na página 37.

CHANDRASHEKAR, G.; SAHIN, F. A survey on feature selection methods. *Computers & Electrical Engineering*, v. 40, n. 1, p. 16–28, 2014. ISSN 0045-7906. Citado na página 44.

CLARK, E.; ARAKI, K. Text normalization in social media: Progress, problems and applications for a pre-processing system of casual english. *Procedia – Social and Behavioral Sciences*, v. 27, p. 2–11, 2011. ISSN 1877-0428. Citado na página 41.

CORTES, C.; VAPNIK, V. N. Support-vector networks. *Machine Learning*, Kluwer Academic Publishers, Hingham, MA, USA, v. 20, n. 3, p. 273–297, set. 1995. Citado 4 vezes nas páginas 23, 30, 73 e 100.

COVER, T. M.; HART, P. E. Nearest neighbor pattern classification. *IEEE Transaction on Information Theory*, IEEE, v. 13, n. 1, p. 21–27, jan. 1967. ISSN 0018-9448. Citado 4 vezes nas páginas 30, 48, 73 e 100.

COVER, T. M.; THOMAS, J. A. *Elements of Information Theory*. 2th. ed. Hoboken, New Jersey: Wiley-Interscience, 2006. Citado 4 vezes nas páginas 47, 48, 49 e 50.

CRAMMER, K.; DREDZE, M.; PEREIRA, F. Confidence-weighted linear classification for text categorization. *Journal of Machine Learning Research*, JMLR.org, v. 13, n. 1, p. 1891–1926, jun. 2012. Citado 2 vezes nas páginas 24 e 32.

CREUTZ, M.; LAGUS, K. Unsupervised discovery of morphemes. In: *Proceedings of the 6th Workshop of the ACL Special Interest Group in Computational Phonology (ACL/-SIGPHON'02)*. Philadelphia, PA, USA: Association for Computational Linguistics, 2002. p. 21–30. Citado na página 55.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, JMLR.org, v. 7, p. 1–30, dez. 2006. Citado 2 vezes nas páginas 77 e 78.

DOMINGOS, P. The role of Occam's razor in knowledge discovery. *Data Mining and Knowledge Discovery*, Kluwer Academic Publishers, Hingham, MA, USA, v. 3, p. 409–425, 1999. Citado na página 22.

DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern Classification*. 2rd. ed. [S.l.]: Wiley-Interscience, 2000. ISBN 0471056693. Citado na página 54.

ESCALANTE, H. J.; GARCÍA-LIMóN, M. A.; MORALES-REYES, A.; GRAFF, M.; GóMEZ, M. M. y; MORALES, E. F.; MARTÍNEZ-CARRANZA, J. Term-weighting learning via genetic programming for text classification. *Knowledge-Based Systems*, Elsevier, Amsterdam, The Netherlands, The Netherlands, v. 83, n. C, p. 176–189, jul. 2015. ISSN 0950-7051. Citado na página 36.

FAYYAD, U. M.; IRANI, K. B. Multi-interval discretization of continuous-valued attributes for classification learning. In: *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI'93)*. Chambéry, France: Morgan Kaufmann, 1993. p. 1022–1029. Citado 2 vezes nas páginas 97 e 101.

FREUND, Y.; SCHAPIRE, R. E. Experiments with a new boosting algorithm. In: *Proceedings of the 13th International Conference on Machine Learning (ICML'96)*. Bari, Italy: Morgan Kaufmann, 1996. p. 148–156. Citado na página 30.

FREUND, Y.; SCHAPIRE, R. E. Large margin classification using the perceptron algorithm. *Machine Learning*, Springer, v. 37, n. 3, p. 277–296, dez. 1999. Citado na página 74.

FRIEDMAN, J. H. On bias, variance, 0/1—loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, Springer, v. 1, n. 1, p. 55–77, 1997. Citado na página 23.

GALAVOTTI, L.; SEBASTIANI, F.; SIMI, M. Experiments on the use of feature selection and negative evidence in automated text categorization. In: *Proceedings of the 4th European Conference on Research and Advanced Technology for Digital Libraries (ECDL'00)*. Lisbon, Portugal: Springer, 2000. p. 59–68. Citado na página 65.

GARCÍA, S.; FERNÁNDEZ, A.; LUENGO, J.; HERRERA, F. A study of statistical techniques and performance measures for genetics-based machine learning: Accuracy and interpretability. *Soft Computing*, Springer-Verlag, Berlin, Heidelberg, v. 13, n. 10, p. 959–977, abr. 2009. Citado na página 78.

GOLDSMITH, J. An algorithm for the unsupervised learning of morphology. *Natural Language Engineering*, Cambridge University Press, New York, NY, USA, v. 12, n. 4, p. 353–371, dez. 2006. Citado na página 55.

GONZÁLEZ-RUBIO, J.; CASACUBERTA, F. Cost-sensitive active learning for computer-assisted translation. *Pattern Recognition Letters*, v. 37, n. 1, p. 124–134, 2014. Citado na página 27.

GOSWAMI, G.; SINGH, R.; VATSA, M. Machine intelligence and signal processing. In: _____. New Delhi: Springer India, 2016. v. 390, cap. Automated Spam Detection in Short Text Messages, p. 85–98. ISBN 978-81-322-2625-3. Citado na página 37.

GRAY, R. M. *Entropy and Information Theory*. New York, London: Springer, 2011. ISBN 978-1-4419-7969-8. Citado na página 47.

GRÜNWALD, P.; KONTKANEN, P.; MYLLYMÄKI, P.; SILANDER, T.; TIRRI, H. Minimum encoding approaches for predictive modeling. In: *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI'98)*. Madison, Wisconsin, USA: Morgan Kaufmann, 1998. p. 183–192. Citado na página 52.

GRÜNWALD, P. D. A tutorial introduction to the minimum description length principle. In: *Advances in Minimum Description Length: Theory and Applications*. [S.l.]: MIT Press, 2005. Citado 2 vezes nas páginas 50 e 53.

GRÜNWALD, P. D. *The Minimum Description Length Principle*. [S.l.]: The MIT Press, 2007. Citado 3 vezes nas páginas 10, 52 e 53.

GRÜNWALD, P. D.; MYUNG, I. J.; PITT, M. A. *Advances in Minimum Description Length: Theory and Applications*. [S.l.]: The MIT Press, 2005. Citado 3 vezes nas páginas 47, 52 e 53.

GUTLEIN, M.; FRANK, E.; HALL, M.; KARWATH, A. Large-scale attribute selection using wrappers. In: *Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Data Mining (CIDM'09)*. Nashville, TN, USA: IEEE, 2009. p. 332–339. Citado na página 37.

HAMIDI, O.; POOROLAJAL, J.; SADEGHIFAR, M.; ABBASI, H.; MARYANAJI, Z.; FARIDI, H. R.; TAPAK, L. A comparative study of support vector machines and artificial neural networks for predicting precipitation in iran. *Theoretical and Applied Climatology*, v. 119, n. 3, p. 723–731, 2015. ISSN 1434-4483. Citado na página 27.

HANSEN, M. H.; YU, B. Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, v. 96, n. 454, p. 746–774, jun. 2001. Citado 2 vezes nas páginas 52 e 53.

HASTIE, T. J.; TIBSHIRANI, R. J.; FRIEDMAN, J. H. *The elements of statistical learning : data mining, inference, and prediction*. 2th. ed. New York, NY, USA: Springer, 2009. ISBN 978-0-387-84857-0. Citado 3 vezes nas páginas 28, 31 e 47.

HAYKIN, S. *Neural Networks: A Comprehensive Foundation*. 2th. ed. New York, NY, USA: Prentice Hall, 1998. Citado na página 30.

HAYKIN, S. *Neural Networks and Learning Machines*. New York, NY, USA: Prentice Hall, 2008. ISBN 9780131471399. Citado na página 23.

HOI, S. C. H.; WANG, J.; ZHAO, P. Libol: A library for online learning algorithms. *Journal of Machine Learning Research*, JMLR.org, v. 15, n. 1, p. 495–499, jan. 2014. Citado na página 74.

HSU, C.-W.; LIN, C.-J. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, v. 13, n. 2, p. 415–425, mar. 2002. Citado 4 vezes nas páginas 24, 29, 30 e 74.

INCE, I.; KLAWONN, F. Handling different levels of granularity within naive Bayes classifiers. In: *Intelligent Data Engineering and Automated Learning – IDEAL 2013*. [S.l.]: Springer Berlin Heidelberg, 2013, (Lecture Notes in Computer Science, v. 8206). p. 521–528. Citado na página 55.

JIANG, L.; WANG, S.; LI, C.; ZHANG, L. Structure extended multinomial naive bayes. *Information Sciences: an International Journal*, Elsevier, New York, NY, USA, v. 329, n. C, p. 346–356, fev. 2016. Citado na página 37.

JOACHIMS, T. Text categorization with suport vector machines: Learning with many relevant features. In: *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*. Chemnitz, Germany: Springer, 1998. p. 137–142. Citado na página 73.

KATZ, G.; OFEK, N.; SHAPIRA, B. Consent: Context-based sentiment analysis. *Knowledge-Based Systems*, v. 84, p. 162–178, 2015. ISSN 0950-7051. Citado na página 37.

KILINÇ, D.; ÖZÇIFT, A.; BOZYIGIT, F.; YILDIRIM, P.; YÜCALAR, F.; BORANDAG, E. Ttc-3600: A new benchmark dataset for turkish text categorization. *Journal of Information Science*, dez. 2015. Citado na página 82.

KIM, S.; KWEON, I. S. Simultaneous classification and visual word selection using entropy-based minimum description length. In: *Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06)*. Hong Kong, China: IEEE, 2006. p. 650–653. Citado na página 56.

KOHAVI, R.; JOHN, G. H. Wrappers for feature subset selection. *Artificial Intelligence*, v. 97, n. 1, p. 273–324, 1997. ISSN 0004-3702. Citado na página 44.

KOHONEN, T. The self-organizing map. In: *Proceedings of the IEEE*. [S.l.: s.n.], 1990. v. 9, n. 78, p. 1464–1480. Citado na página 31.

KOHONENKO, I. The minimum description length based decision tree pruning. In: *PRICAI'98: Topics in Artificial Intelligence*. [S.l.]: Springer Berlin Heidelberg, 1998, (Lecture Notes in Computer Science, v. 1531). p. 228–237. Citado na página 54.

KOLMOGOROV, A. N. Three approaches to the quantitative definition of information. *Problems of Information Transmission*, v. 1, p. 1–7, 1965. Citado 2 vezes nas páginas 22 e 52.

KUMAR, S.; GAO, X.; WELCH, I.; MANSOORI, M. A machine learning based web spam filtering approach. In: *Proceedings of the 30th IEEE International Conference on Advanced Information Networking and Applications (AINA'16)*. Crans-Montana, Switzerland: IEEE, 2016. p. 973–980. ISSN 1550-445X. Citado na página 37.

KUMAR, V.; MINZ, S. Feature selection: A literature review. *Smart Computing Review*, v. 4, n. 3, p. 211–229, 2014. Citado na página 44.

LAM, W.; BACCHUS, F. Learning Bayesian belief networks: an approach based on the MDL principle. *Computational Intelligence*, Blackwell Publishing Ltd, Cambridge, MA, USA, v. 10, n. 3, p. 269–293, ago. 1994. Citado na página 54.

LAMIREL, J.; CUXAC, P.; CHIVUKULA, A. S.; HAJLAOUI, K. Optimizing text classification through efficient feature selection based on quality metric. *Journal of Intelligent Information Systems*, Kluwer Academic Publishers, Hingham, MA, USA, v. 45, n. 3, p. 379–396, dez. 2015. ISSN 0925-9902. Citado na página 44.

LEWIS, D. D.; RINGUETTE, M. A comparison of two learning algorithms for text categorization. In: *Proceedings of the 3rd annual symposium on document analysis and information retrieval (SDAIR'94)*. Las Vegas, NV: [s.n.], 1994. v. 33, p. 81–93. Citado 2 vezes nas páginas 73 e 82.

LI, J.; MOUCHèRE, H.; VIARD-GAUDIN, C. An annotation assistance system using an unsupervised codebook composed of handwritten graphical multi-stroke symbols. *Pattern Recognition Letters*, Elsevier, New York, NY, USA, v. 35, p. 46–57, jan. 2014. Citado na página 55.

LI, M.; VITÁNYI, P. M. *An Introduction to Kolmogorov Complexity and Its Applications*. 3rd. ed. New York, NY: Springer, 2008. Citado 2 vezes nas páginas 22 e 52.

LI, Y.; LONG, P. M. The relaxed online maximum margin algorithm. *Machine Learning*, Kluwer Academic Publishers, Hingham, MA, USA, v. 46, n. 1-3, p. 361–387, jan. 2002. Citado 2 vezes nas páginas 32 e 74.

LIU, H.; MOTODA, H. *Feature Selection for Knowledge Discovery and Data Mining*. Norwell, MA, USA: Kluwer Academic Publishers, 1998. ISBN 079238198X. Citado na página 44.

LIU, Y.; LOH, H. T.; SUN, A. Imbalanced text classification: A term weighting approach. *Expert Systems with Applications*, v. 36, n. 1, p. 690–701, jan. 2009. ISSN 0957–4174. Citado na página 36.

MACKAY, D. J. C. *Information Theory, Inference & Learning Algorithms*. New York, NY, USA: Cambridge University Press, 2005. ISBN 0521642981. Citado 3 vezes nas páginas 47, 48 e 54.

MANNING, C. D.; RAGHAVAN, P.; SCHüTZE, H. *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2009. Citado 6 vezes nas páginas 36, 38, 39, 40, 60 e 73.

MCCALLUM, A.; NIGAM, K. A comparison of event models for naive Bayes text classification. In: *Proceedings of the 15th AAAI Workshop on Learning for Text Categorization*

(AAAI'98). Madison, Wisconsin: [s.n.], 1998. p. 41–48. Citado 4 vezes nas páginas 30, 73, 74 e 100.

MEHTA, M.; AGRAWAL, R.; RISSANEN, J. SLIQ: A fast scalable classifier for data mining. In: *Advances in Database Technology – EDBT'96*. [S.l.]: Springer Berlin Heidelberg, 1996, (Lecture Notes in Computer Science, v. 1057). p. 18–32. Citado na página 54.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G.; DEAN, J. Distributed representations of words and phrases and their compositionality. In: *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13)*. Lake Tahoe, Nevada, USA: Curran Associates Inc., 2013. p. 3111–3119. Citado na página 42.

MISHNE, G.; CARMEL, D.; LEMPEL, R. Blocking blog spam with language model disagreement. In: *Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web (AIRWeb'05)*. Chiba, Japan: [s.n.], 2005. Citado na página 92.

MITCHELL, T. M. *Machine Learning*. 1. ed. New York, NY, USA: McGraw-Hill, 1997. ISBN 0070428077, 9780070428072. Citado 3 vezes nas páginas 27, 29 e 54.

MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. *Foundations of machine learning*. Cambridge, Massachusetts, USA: MIT press, 2012. Citado na página 27.

MUHAMMAD, A.; WIRATUNGA, N.; LOTHIAN, R. Contextual sentiment analysis for social media genres. *Knowledge-Based Systems*, p. 1–10, 2016. ISSN 0950-7051. Citado na página 37.

MURPHY, K. P. *Machine Learning: A Probabilistic Perspective*. [S.l.]: The MIT Press, 2012. ISBN 9780262018029. Citado na página 28.

NG, H. T.; GOH, W. B.; LOW, K. L. Feature selection, perceptron learning, and a usability case study for text categorization. In: *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'97)*. Philadelphia, PA, USA: ACM, 1997. p. 67–73. Citado na página 65.

ORENGO, V. M.; HUYCK, C. R. A stemming algorithm for the portuguese language. In: *Proceedings of the 8th International Symposium on String Processing and Information Retrieval (SPIRE'01)*. Laguna de San Raphael, Chile: IEEE, 2001. p. 183–193. Citado na página 40.

OTT, M.; CARDIE, C.; HANCOCK, J. T. Negative deceptive opinion spam. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL/HLT'11)*. Atlanta, Georgia, USA: Association for Computational Linguistics, 2013. p. 01–05. Citado na página 93.

OTT, M.; CHOI, Y.; CARDIE, C.; HANCOCK, J. T. Finding deceptive opinion spam by any stretch of the imagination. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL/HLT'11)*. Portland, Oregon: Association for Computational Linguistics, 2011. p. 309–319. Citado na página 93.

PAPADAKIS, G.; GIANNAKOPOULOS, G.; PALIOURAS, G. Graph vs. bag representation models for the topic classification of web documents. *World Wide Web*, Kluwer Academic Publishers, Hingham, MA, USA, v. 19, n. 5, p. 887–920, set. 2016. ISSN 1386-145X. Citado na página 42.

PORTER, M. F. An algorithm for suffix stripping. *Program*, MCB UP Ltd, v. 14, n. 3, p. 130–137, 1980. Citado na página 40.

PORWAL, U.; SHI, Z.; SETLUR, S. Machine learning in handwritten arabic text recognition. *Handbook of Statistics*, v. 31, p. 443–469, 2013. Citado na página 27.

POTAPOV, A. S. Principle of representational minimum description length in image analysis and pattern recognition. *Pattern Recognition and Image Analysis*, Springer, v. 22, n. 1, p. 82–91, mar. 2012. Citado na página 55.

PRINCIPE, J. C. *Information Theoretic Learning: Renyi's Entropy and Kernel Perspectives*. 1st. ed. [S.l.]: Springer, 2010. ISBN 1441915699, 9781441915696. Citado 2 vezes nas páginas 47 e 48.

QUINLAN, J. R. Induction of decision trees. *Machine Learning*, Kluwer Academic Publishers, Hingham, MA, USA, v. 1, n. 1, p. 81–106, mar. 1986. Citado na página 30.

QUINLAN, J. R. Improved use of continuous attributes in c4.5. *Journal of Artificial Intelligence Research*, AI Access Foundation, USA, v. 4, n. 1, p. 77–90, mar. 1996. Citado na página 30.

QUINLAN, J. R.; RIVEST, R. L. Inferring decision trees using the minimum description length principle. *Information and Computation*, Academic Press, Inc., Duluth, MN, USA, v. 80, n. 3, p. 227–248, mar. 1989. Citado na página 54.

RENNIE, J. D.; SHIH, L.; TEEVAN, J.; KARGER, D. R. Tackling the poor assumptions of naive Bayes text classifiers. In: *Proceedings of the 20th International Conference on Machine Learning (ICML'03)*. Washington, DC, USA: AAAI Press, 2003. v. 3, p. 616–623. Citado 2 vezes nas páginas 43 e 59.

RISSANEN, J. Modeling by shortest data description. *Automatica*, v. 14, n. 5, p. 465–471, set. 1978. Citado 5 vezes nas páginas 22, 47, 51, 52 e 53.

RISSANEN, J. A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, v. 11, n. 2, p. 416–431, jun. 1983. Citado 3 vezes nas páginas 51, 52 e 53.

RISSANEN, J. Fisher information and stochastic complexity. *IEEE Transaction on Information Theory*, v. 42, n. 1, p. 40–47, jan. 1996. Citado 2 vezes nas páginas 53 e 58.

ROBINET, V.; LEMAIRE, B.; GORDON, M. B. MDLChunker: A MDL-based cognitive model of inductive learning. *Cognitive Science*, Wiley Online Library, v. 35, n. 7, p. 1352–1389, set.–out. 2011. Citado na página 56.

ROCCHIO, J. J. Relevance feedback in information retrieval. In: SALTON, G. (Ed.). *The Smart retrieval system - experiments in automatic document processing*. Englewood Cliffs, NJ: Prentice-Hall, 1971. p. 313–323. Citado 3 vezes nas páginas 30, 60 e 73.

ROGATI, M.; YANG, Y. High-performing feature selection for text classification. In: *Proceedings of the 11th International Conference on Information and Knowledge Management (CIKM'02)*. McLean, Virginia, USA: ACM, 2002. p. 659–661. ISBN 1-58113-492-4. Disponível em: ⟨http://doi.acm.org/10.1145/584792.584911⟩. Citado na página 82.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, n. 6, p. 386–408, 1958. Citado na página 32.

ROSSI, R. G.; FALEIROS, T. de P.; LOPES, A. de A.; REZENDE, S. O. Inductive model generation for text categorization using a bipartite heterogeneous network. In: *Proceedings of the IEEE 12th International Conference on Data Mining (ICDM'12)*. Brussels, Belgium: IEEE, 2012. p. 1086–1091. Citado na página 37.

ROSSI, R. G.; LOPES, A. de A.; REZENDE, S. O. Optimization and label propagation in bipartite heterogeneous networks to improve transductive classification of texts. *Information Processing & Management*, Elsevier, v. 52, n. 2, p. 217–257, mar. 2016. Citado 6 vezes nas páginas 27, 37, 42, 70, 72 e 73.

ROSSI, R. G.; MARCACINI, R. M.; REZENDE, S. O. *Benchmarking Text Collections for Classification and Clustering Tasks*. [S.l.], 2013. Citado na página 72.

RUSSELL, S. J.; NORVIG, P. *Artificial Intelligence: A Modern Approach*. 3rd. ed. [S.l.]: Prentice Hall, 2010. Citado na página 54.

SAIF, H.; HE, Y.; ALANI, H. Alleviating data sparsity for twitter sentiment analysis. In: *Proceedings of the 2nd Workshop on Making Sense of Microposts (MSM'12)*. Lyon, France: CEUR-WS.org, 2012. v. 838, p. 2–9. Citado na página 82.

SALTON, G.; WONG, A.; YANG, C. S. A vector space model for automatic indexing. *Magazine Communications of the ACM*, ACM, New York, NY, USA, v. 18, n. 11, p. 613–620, nov. 1975. ISSN 0001-0782. Citado na página 43.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM JOURNAL OF RESEARCH AND DEVELOPMENT*, p. 71–105, 1959. Citado na página 27.

SEBASTIANI, F. Machine learning in automated text categorization. *ACM Computing Surveys*, ACM, New York, NY, USA, v. 34, n. 1, p. 1–47, mar. 2002. Citado 4 vezes nas páginas 38, 42, 44 e 65.

SHANNON, C. E. A mathematical theory of communication. *The Bell System Technical Journal*, v. 27, p. 379–423, 623–656, jul. 1948. Citado 2 vezes nas páginas 48 e 53.

SMART, W. D.; KAELBLING, L. P. Effective reinforcement learning for mobile robots. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'02)*. Washington, DC, USA: IEEE, 2002. v. 4, p. 3404–3410. Citado na página 31.

SOKOLOVA, M.; LAPALME, G. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, Pergamon Press, Tarrytown, NY, USA, v. 45, n. 4, p. 427–437, jul. 2009. Citado 3 vezes nas páginas 33, 35 e 36.

SOLOMONOFF, R. J. A formal theory of inductive inference: Parts 1 & 2. *Information and Control*, v. 7, n. 1, 2, p. 1–22, 224–254, mar., jun. 1964. Citado na página 52.

TABUS, I.; ASTOLA, J. On the use of mdl principle in gene expression prediction. *EURASIP Journal on Applied Signal Processing*, Hindawi Publishing Corp., New York, NY, United States, v. 2001, n. 4, p. 297–303, dez. 2001. Citado na página 55.

TANG, J.; ALELYANI, S.; LIU, H. Feature selection for classification: A review. In: AGGARWAL, C. C. (Ed.). *Data Classification: Algorithms and Applications*. [S.l.]: CRC Press, 2014. p. 37–64. Citado na página 45.

TATAW, O. M.; RAKTHANMANON, T.; KEOGH, E. J. Clustering of symbols using minimal description length. In: *Proceedings of the 12th International Conference on Analysis and Recognition (ICDAR'13)*. Washington, DC, USA: IEEE, 2013. p. 180–184. Citado na página 55.

TSENG, C.-J.; LU, C.-J.; CHANG, C.-C.; CHEN, G.-D. Application of machine learning to predict the recurrence-proneness for cervical cancer. *Neural Computing and Applications*, v. 24, n. 6, p. 1311–1316, 2014. Citado na página 27.

TURIAN, J.; RATINOV, L.; BENGIO, Y. Word representations: A simple and general method for semi-supervised learning. In: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL'10)*. Uppsala, Sweden: Association for Computational Linguistics, 2010. p. 384–394. Citado na página 42.

UYSAL, A. K.; GUNAL, S. A novel probabilistic feature selection method for text classification. *Knowledge-Based Systems*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 36, p. 226–235, dez. 2012. ISSN 0950-7051. Citado 2 vezes nas páginas 37 e 65.

UYSAL, A. K.; GUNAL, S. The impact of preprocessing on text classification. *Information Processing & Management*, Pergamon Press, Inc., Tarrytown, NY, USA, v. 50, n. 1, p. 104–112, jan. 2014. ISSN 0306-4573. Citado 2 vezes nas páginas 39 e 40.

UYSAL, A. K.; GUNAL, S.; ERGIN, S.; GUNAL, E. S. A novel framework for sms spam filtering. In: *Proceedings of the 2012 International Symposium on Innovations in Intelligent Systems and Applications (INISTA'12)*. Trabzon, Turkey: IEEE, 2012. p. 1–4. Citado na página 37.

VO, D.; OCK, C. Learning to classify short text from scientific documents using topic models with various types of knowledge. *Expert Systems with Applications*, Elsevier, v. 42, n. 3, p. 1684–1698, fev. 2015. ISSN 0957-4174. Citado 2 vezes nas páginas 37 e 41.

WALLACE, C. S.; BOULTON, D. M. An information measure for classification. *The Computer Journal*, v. 11, n. 2, p. 185–194, ago. 1968. Citado na página 52.

WALLACE, C. S.; FREEMAN, P. R. Estimation and inference by compact coding. *Journal of the Royal Statistical Society*, v. 49, n. 3, p. 240–265, 1987. Citado na página 52.

WEI, J.; JIAN-QI, Z.; XIANG, Z. Face recognition method based on support vector machine and particle swarm optimization. *Expert Systems with Applications*, v. 38, n. 4, p. 4390–4393, 2011. Citado na página 27.

WEISS, S.; INDURKHYA, N.; ZHANG, T.; DAMERAU, F. *Text Mining: Predictive Methods for Analyzing Unstructured Information*. [S.l.]: Springer Verlag, 2004. ISBN 0387954333. Citado 3 vezes nas páginas 38, 40 e 42.

WILBUR, W. J.; KIM, W. The ineffectiveness of within-document term frequency in text classification. *Information Retrieval*, Kluwer Academic Publishers, Hingham, MA, USA, v. 12, n. 5, p. 509–525, out. 2009. Citado na página 43.

WITTEN, I. H.; FRANK, E.; HALL, M. A. *Data Mining: Practical Machine Learning Tools and Techniques*. 3rd. ed. San Francisco, CA, USA: Morgan Kaufmann, 2011. Citado 2 vezes nas páginas 23 e 33.

WU, Q.; YE, Y.; ZHANG, H.; NG, M. K.; HO, S. Forestexter: An efficient random forest algorithm for imbalanced text categorization. *Knowledge-Based Systems*, Elsevier, Amsterdam, The Netherlands, The Netherlands, v. 67, p. 105–116, set. 2014. ISSN 0950-7051. Citado 3 vezes nas páginas 27, 37 e 73.

WU, X.; KUMAR, V.; QUINLAN, J. R.; GHOSH, J.; YANG, Q.; MOTODA, H.; MCLACHLAN; NG, A.; LIU, B.; YU, P. S.; ZHOU, Z.; STEINBACH, M.; HAND, D. J.; STEINBERG, D. Top 10 algorithms in data mining. *Knowledge and Information Systems*, Springer-Verlag, New York, NY, USA, v. 14, n. 1, p. 1–37, dez. 2008. Citado na página 73.

WU, X.; YU, K.; DING, W.; WANG, H.; ZHU, X. Online feature selection with streaming features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 35, n. 5, p. 1178–1192, May 2013. Citado na página 45.

WU, X.; YU, K.; WANG, H.; DING, W. Online streaming feature selection. In: FÜRNKRANZ, J.; JOACHIMS, T. (Ed.). *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*. Haifa, Israel: Omnipress, 2010. p. 1159–1166. Disponível em: ⟨http://www.icml2010.org/papers/238.pdf⟩. Citado na página 45.

YANG, Y. An evaluation of statistical approaches to text categorization. *Information Retrieval*, Kluwer Academic Publishers, Hingham, MA, USA, v. 1, n. 1–2, p. 69–90, maio 1999. ISSN 1386-4564. Citado na página 36.

YANG, Y.; PEDERSEN, J. O. A comparative study on feature selection in text categorization. In: *Proceedings of the 14th International Conference on Machine Learning (ICML'97)*. Nashville, TN, USA: Morgan Kaufmann, 1997. p. 412–420. Citado 2 vezes nas páginas 65 e 82.

YU, B.; XU, Z. ben. A comparative study for content-based dynamic spam classification using four machine learning algorithms. *Knowledge-Based Systems*, Elsevier, Amsterdam, The Netherlands, v. 21, n. 4, p. 355–362, maio 2008. Citado 2 vezes nas páginas 37 e 73.

ZHANG, H.; ZHONG, G. Improving short text classification by learning vector representations of both words and hidden topics. *Knowledge-Based Systems*, v. 102, p. 76–86, jun. 2016. ISSN 0950-7051. Citado na página 36.

ZHANG, L.; JIANG, L.; LI, C.; KONG, G. Two feature weighting approaches for naive bayes text classifiers. *Knowledge-Based Systems*, Elsevier, v. 100, p. 137–144, 2016. Citado na página 70.

ZHANG, T. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In: *Proceedings of the 21th International Conference on Machine Learning (ICML'04)*. Banff, Alberta, Canada: ACM, 2004. p. 116–123. Citado 2 vezes nas páginas 32 e 74.

ZHANG, X.; LIU, X.; WANG, Z. J. Evaluation of a set of new orf kernel functions of svm for speech recognition. *Engineering Applications of Artificial Intelligence*, v. 26, n. 10, p. 2574–2580, 2013. Citado na página 27.

ZINKEVICH, M. Online convex programming and generalized infinitesimal gradient ascent. In: *Proceedings of the 20th International Conference on Machine Learning (ICML'03)*. Washington, DC, USA: AAAI Press, 2003. p. 928–936. Citado 2 vezes nas páginas 32 e 74.

APÊNDICE A – Aplicação do MDLText na classificação de documentos de texto curtos e ruidosos com o auxílio de técnicas de normalização léxica e indexação semântica

Classificação de documentos de texto curtos e ruidosos é uma tarefa que vem ganhando cada vez mais notoriedade no mundo atual. Usuários da Web têm gerado um grande volume de textos nas redes sociais, blogs, sites de notícias, sites de vendas e aplicativos de *smartphones*, por meio de comentários e mensagens instantâneas. Esses textos precisam ser analisados para prover segurança aos usuários, facilitar o estudo do comportamento das pessoas, agilizar a detecção de movimentos culturais, melhorar os serviços prestados pelas empresas por meio da análise dos comentários dos usuários sobre seus produtos e seu atendimento, entre outras aplicações.

Categorizar documentos de texto curtos é uma tarefa desafiadora devido ao baixo número de termos existentes no texto, o que consequentemente faz com que a representação torne-se muito esparsa e pode resultar na redução drástica do desempenho dos métodos tradicionais de classificação. Somado a isso, tais documentos costumam ser repletos de gírias, símbolos e abreviações que dificultam até mesmo a extração dos atributos e a geração do modelo espaço-vetorial para ser usado no processo de classificação (ALMEIDA *et al.*, 2016).

Nesse cenário, o objetivo do estudo apresentado neste apêndice é analisar o desempenho do `MDLText` na classificação de documentos de texto curtos e ruidosos. Adicionalmente, é analisado se o uso de técnicas de normalização léxica e indexação semântica podem ser benéficas para a classificação.

O cenário considerado neste apêndice é o de aprendizado *online*, pois muitas aplicações que envolvem textos curtos, tais como redes sociais e aplicativos de mensagens instantâneas, precisam de modelos de predição dinâmicos, que possam ser atualizados constantemente. Métodos de aprendizado *offline*, cujos modelos de predição são estáticos, não são apropriados para este problema.

Inicialmente são apresentados os principais conceitos sobre normalização léxica e indexação semântica. Em seguida, é proposta uma nova técnica que combina as predições obtidas pelos métodos de classificação usando documentos de texto originais e suas variações obtidas após aplicar normalização e incluir informações semânticas. Posteriormente, são detalhadas as bases de dados, métodos, medidas de desempenho e outras in-

formações importantes usadas nos experimentos. Por fim, são apresentados os resultados e conclusões.

A.1 Normalização do texto e indexação semântica

Os documentos de texto gerados por usuários da Web e de *smartphones*, são geralmente curtos e repletos de gírias, expressões idiomáticas, símbolos, *emoticons* e abreviações. Tais características degradam o desempenho dos métodos de classificação de texto tradicionais. Porém, em um estudo recente, Almeida *et al.* (2016) demonstraram que os métodos de classificação podem ter seu desempenho melhorado quando são auxiliados por técnicas de normalização léxica e indexação semântica. Os autores propuseram a ferramenta `TextExpansion`[1], que é um *framework* para normalização e expansão de textos curtos.

Basicamente, métodos de expansão combinam técnicas estado-da-arte de normalização léxica e detecção de contexto usando dicionários semânticos. Cada exemplo de texto puro (sem pré-processamento) é processado em três diferentes estágios, cada um gerando uma nova representação em ciclos (ALMEIDA *et al.*, 2016): normalização léxica, indexação semântica e desambiguação (consulte a Seção 3.1.2).

Na `TextExpansion`, a normalização léxica emprega o dicionário NoSlang[2] para converter gírias, símbolos, abreviações e acrônimos para sua forma canônica na língua inglesa. A indexação semântica e desambiguação usam o repositório semântico LDB BabelNet[3].

A `TextExpansion` expande a amostra textual em três estágios: normalização, indexação semântica e desambiguação. Então, dada uma regra de expansão, as amostras resultantes em cada estágio são unidas para formar um único exemplo que pode ser processado por uma técnica de aprendizado de máquina. A Figura 28 ilustra o processo.

Após processar um documento de texto usando a `TextExpansion`, dez novas versões expandidas desse documento podem ser geradas, uma para cada possível regra de expansão, definida pela combinação de quatro parâmetros: (1) manter os termos originais, (2) aplicar normalização léxica, (3) obter conceitos por indexação semântica e (4) aplicar desambiguação. Todas as regras de expansão são apresentadas na Tabela 19. Para maiores detalhes, consulte Almeida *et al.* (2016).

[1] A ferramenta `TextExpansion` está disponível em: ⟨http://lasid.sor.ufscar.br/expansion⟩. Acessado em: 19/01/2017.

[2] O dicionário NoSlang está disponível em: ⟨http://www.noslang.com/dictionary/full/⟩. Acessado em: 19/01/2017.

[3] O repositório LDB BabelNet está disponível em: ⟨http://babelnet.org/⟩. Acessado em: 19/01/2017.

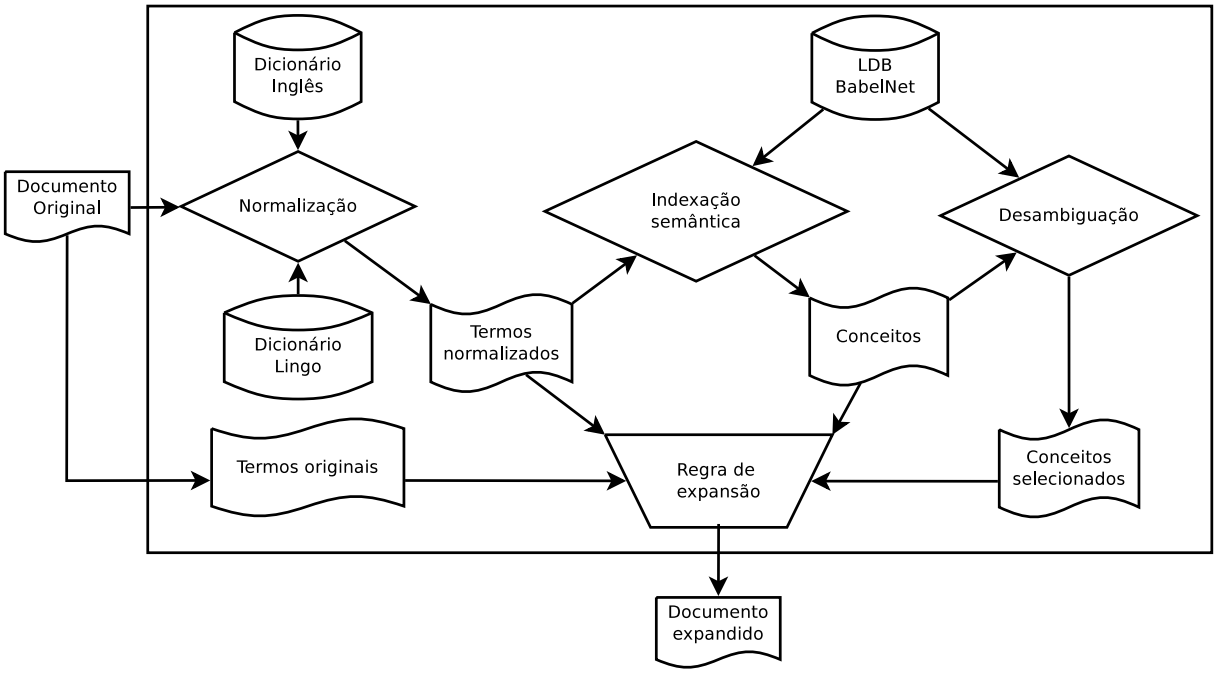


Figura 28 – O documento de texto original é processado por dicionários semânticos e técnicas de detecção de contexto, criando-se novos documentos expandidos e normalizados. Dada uma regra de expansão, os documentos normalizados e expandidos são unidos formando um documento final (Fonte: Almeida *et al.* (2016)).

Tabela 19 – Possíveis regras de expansão utilizadas na ferramenta `TextExpansion`.

	Os termos originais foram mantidos?	Foi aplicada normalização?	Foi aplicada indexação semântica?	Foi aplicada desambiguação?
Expansão 1	X		X	
Expansão 2	X			X
Expansão 3	X	X		
Expansão 4	X	X	X	
Expansão 5	X	X		X
Expansão 6			X	
Expansão 7				X
Expansão 8		X		
Expansão 9		X	X	
Expansão 10		X		X

A.1.1 Combinação das predições obtidas por diferentes regras de expansão

Dado que é possível criar dez novos documentos de texto para cada documento original, pode ser feita uma combinação das predições obtidas pelos métodos de classificação usando cada um deles como entrada, em vez de usá-los individualmente. Assim, esta tese apresenta uma técnica *ensemble* que combina as predições individuais obtidas usando o documento de texto original e os documentos gerados pela `TextExpansion` (Figura 29).

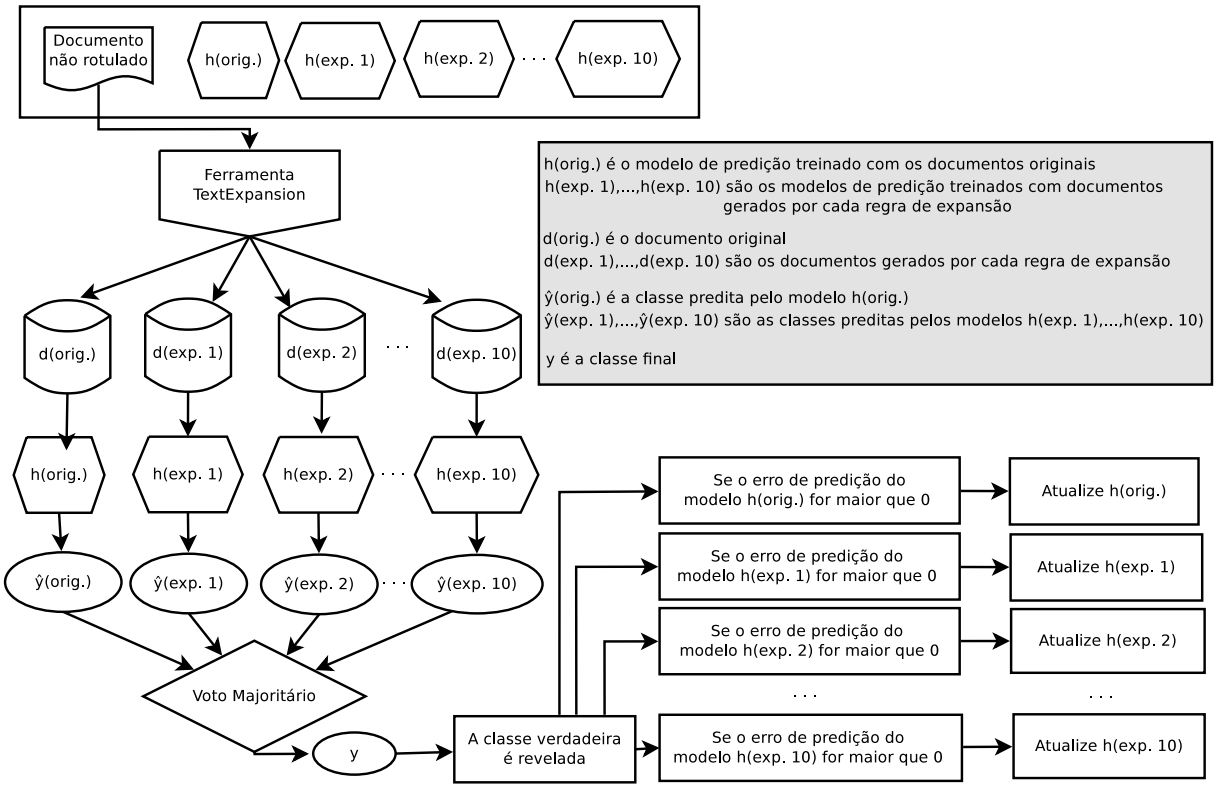


Figura 29 – Técnica *ensemble* que combina os classificadores usando os documentos originais e os documentos gerados pelas regras de expansão.

Conforme ilustrado na Figura 29, há um modelo preditivo gerado usando os documentos de treinamento originais e dez outros modelos preditivos gerados usando os documentos pré-processados pela ferramenta `TextExpansion` (um modelo preditivo para cada regra de expansão apresentada na Tabela 19). Estes modelos de predição podem ser gerados usando qualquer método de classificação *online*. Quando um documento de texto com classe desconhecida é apresentado, cada modelo preditivo faz uma predição. Então, a classe final que será atribuída ao documento é aquela que foi mais votada pelos modelos. Se o erro de predição de um modelo preditivo for maior que zero, então a classe verdadeira do documento é apresentada ao modelo de predição e ele é atualizado.

Como é de se esperar, a técnica *ensemble* demanda um alto custo computacional, pois ela cria e atualiza onze modelos preditivos simultaneamente. Porém, este problema pode ser minimizado se os modelos preditivos forem processados paralelamente.

A.2 Metodologia

Para simular cenários reais, neste estudo foram feitos experimentos considerando um cenário *online*, similar ao que foi apresentado na Seção 2.3. Para a técnica *ensemble*, o processo de classificação *online* foi adaptado para corresponder ao fluxograma apresentado na Figura 29.

Os resultados obtidos pelo `MDLText` no cenário apresentado neste apêndice foram comparados aos resultados obtidos pelos métodos apresentados na Seção 6.1.4, usando as mesmas configurações. As bases de dados usadas nos experimentos foram as mesmas apresentas na Seção 6.2.2.3.

A.2.1 Pré-processamento e *tokenização*

Em todos os experimentos, os textos foram convertidos para letras minúsculas e, depois, os caracteres não-alfanuméricos foram usados como delimitadores no processo de *tokenização*.

Para cada base de dados, a ferramenta `TextExpansion` foi usada para criar dez bases expandidas. Depois, para cada método de classificação, foram realizados os seguintes experimentos:

1. usando apenas os documentos originais;
2. usando os documentos expandidos individualmente;
3. usando o documento original e os documentos expandidos combinados pela técnica *ensemble*.

A Tabela 20 sumariza as principais estatísticas das bases de dados originais e daquelas geradas por cada regra de expansão mostrada na Tabela 19, onde $|\mathcal{T}|$ corresponde ao número de termos (tamanho do vocabulário), enquanto ψ e IQR são, respectivamente, o número de termos por documentos e a amplitude interquartil do número de termos por documento. Apesar da quantidade de exemplos por classe ser a mesma para as bases de dados originais e expandidas, o tamanho do vocabulário, a mediana e o IQR variam bastante.

É interessante notar que a regra de expansão 4 gerou o maior vocabulário e consequentemente a maior mediana e IQR de termos por documento. Por outro lado, para a maioria das bases de dados, a expansão 8 gerou o menor tamanho de vocabulário e a menor mediana e IQR. Quanto às bases de dados originais, os exemplos da base de dados *Review Spam Collection* são as maiores, com uma mediana de 84 termos por documento, enquanto as bases de dados SMS e YouTube tem uma mediana de 11 termos por documento.

A.3 Resultados

A Tabela 21 apresenta as médias da macro F-medida obtidas em dez rodadas dos experimentos com aprendizado *online* usando o processo descrito no Algoritmo 6.1. Para cada método e base de dados avaliados são apresentados os resultados obtidos com

Tabela 20 – Estatísticas das bases de dados originais e expandidas.

	SMS			Blog			Review			YSC - Eminem		
	Tamanho das classes			Tamanho das classes			Tamanho das classes			Tamanho das classes		
	747		4.827	692		332	800		800	245		203
	$\|\mathcal{T}\|$	ψ	IQR	$\|\mathcal{T}\|$	ψ	IQR	$\|\mathcal{T}\|$	ψ	IQR	$\|\mathcal{T}\|$	ψ	IQR
Base de dados original	8.706	11,00	13,00	8.163	22,00	35,50	9.571	84,00	51,00	1.601	7,00	13,50
Expansão 1	16.798	43,00	56,00	16.645	88,00	171,00	18.835	333,00	221,00	4.953	37,00	72,00
Expansão 2	9.897	14,00	17,00	9.352	28,00	46,50	11.004	107,00	64,00	1.997	9,00	16,50
Expansão 3	9.201	14,00	15,00	8.814	27,00	40,50	10.356	102,00	61,00	1.773	9,00	16,00
Expansão 4	17.041	46,00	58,00	17.015	92,00	172,00	19.304	341,00	224,00	5.038	37,00	74,00
Expansão 5	10.326	17,00	20,00	9.934	33,00	51,00	11.682	125,00	75,00	2.155	10,00	21,00
Expansão 6	14.610	42,00	54,00	14.598	85,00	165,00	15.554	315,00	209,00	4.564	37,00	72,00
Expansão 7	7.236	12,00	14,00	6.711	23,00	34,00	6.714	82,00	46,50	1.461	8,00	15,00
Expansão 8	7.223	12,00	13,00	6.931	22,00	33,00	7.292	82,00	47,00	1.424	8,00	14,00
Expansão 9	15.252	43,00	55,00	15.302	87,00	167,50	16.449	323,00	213,50	4.721	37,00	73,00
Expansão 10	8.381	15,00	17,00	8.080	29,00	45,00	8.661	105,00	62,00	1.810	9,00	17,50

	YSC - Katy Perry			YSC - LMFAO			YSC - Psy			YSC - Shakira		
	Tamanho das classes			Tamanho das classes			Tamanho das classes			Tamanho das classes		
	175		175	236		202	175		175	174		196
	$\|\mathcal{T}\|$	ψ	IQR	$\|\mathcal{T}\|$	ψ	IQR	$\|\mathcal{T}\|$	ψ	IQR	$\|\mathcal{T}\|$	ψ	IQR
Base de dados original	1.755	11,00	12,00	954	6,00	5,00	1.429	9,00	9,00	1.357	6,00	12,00
Expansão 1	4.977	39,00	57,00	3.174	33,50	25,00	4.100	38,50	50,00	4.211	23,00	54,00
Expansão 2	2.109	13,00	14,00	1.192	7,00	6,00	1.734	12,00	13,00	1.692	7,00	14,00
Expansão 3	1.896	13,00	14,00	1.068	6,00	6,00	1.574	11,50	13,00	1.501	7,00	12,00
Expansão 4	5.035	40,00	57,00	3.227	34,00	25,00	4.195	41,00	51,00	4.283	23,50	56,00
Expansão 5	2.260	15,00	17,00	1.098	7,00	7,00	1.871	14,00	16,00	1.823	8,00	16,00
Expansão 6	4.631	38,00	55,00	2.973	33,00	24,00	3.834	38,00	49,00	3.863	22,00	52,00
Expansão 7	1.666	12,00	13,00	932	6,00	5,00	1.354	10,50	11,00	1.033	6,00	12,00
Expansão 8	1.604	11,00	12,00	889	6,00	5,00	1.307	10,00	11,00	1.023	6,00	12,00
Expansão 9	4.768	38,00	55,00	3.061	33,00	25,00	3.953	38,00	49,00	4.025	23,00	53,00
Expansão 10	1.974	14,00	16,00	1.101	7,00	7,00	1.610	13,00	14,00	1.551	7,50	14,00

(1) os documentos originais, (2) os documentos expandidos em que a melhor macro F-medida foi obtida e (3) a técnica *ensemble*. Os resultados estão ordenados pela macro F-medida. Os valores em negrito indicam o melhor resultado para cada base de dados.

Tabela 21 – Resultados médios obtidos pelos métodos em 10 rodadas da classificação *on-line* de documentos de texto curtos e ruidosos. Os resultados estão ordenados pela macro F-medida e os valores em negrito indicam o melhor resultado para cada base de dados (**Continua**).

	Base de dados original	Expansão textual	Ensemble
	SMS		
SGD	0,928	0,935	**0,961**
Perceptron	0,917	0,929	0,955
M.NB	0,955	0,952	0,888
`MDLText`	0,949	0,950	0,905
MDL	0,921	0,929	0,884
B.NB	0,892	0,925	0,912

Tabela 21 – Resultados médios obtidos pelos métodos em 10 rodadas da classificação *on-line* de documentos de texto curtos e ruidosos. Os resultados estão ordenados pela macro F-medida e os valores em negrito indicam o melhor resultado para cada base de dados (**Continuação**).

	Base de dados original	Expansão textual	Ensemble
ROMMA	0,902	0,901	0,860
OGD	0,718	0,784	0,741
Blog			
M.NB	**0,858**	0,849	0,838
`MDLText`	0,841	0,843	0,815
SGD	0,809	0,810	0,836
MDL	0,819	0,828	0,801
Perceptron	0,784	0,786	0,822
OGD	0,815	0,801	0,766
B.NB	0,803	0,815	0,808
ROMMA	0,762	0,772	0,743
Review			
B.NB	0,843	0,842	**0,850**
M.NB	0,841	0,849	0,846
SGD	0,804	0,809	0,848
`MDLText`	0,835	0,842	0,792
Perceptron	0,789	0,783	0,838
OGD	0,802	0,791	0,747
MDL	0,770	0,787	0,752
ROMMA	0,712	0,733	0,701
YSC - Eminem			
SGD	0,894	0,890	**0,918**
B.NB	0,913	0,909	0,905
Perceptron	0,874	0,887	0,912
`MDLText`	0,907	0,911	0,868
MDL	0,887	0,903	0,874
M.NB	0,891	0,890	0,901
ROMMA	0,862	0,852	0,808
OGD	0,790	0,798	0,765
YSC - Katy Perry			
B.NB	0,911	0,915	**0,926**
SGD	0,873	0,882	0,920
`MDLText`	0,902	0,917	0,880
M.NB	0,900	0,904	0,905
Perceptron	0,865	0,869	0,895
MDL	0,867	0,886	0,833
OGD	0,855	0,866	0,830
ROMMA	0,847	0,860	0,801
YSC - LMFAO			
SGD	0,901	0,904	**0,930**

Tabela 21 – Resultados médios obtidos pelos métodos em 10 rodadas da classificação *online* de documentos de texto curtos e ruidosos. Os resultados estão ordenados pela macro F-medida e os valores em negrito indicam o melhor resultado para cada base de dados (**Conclusão**).

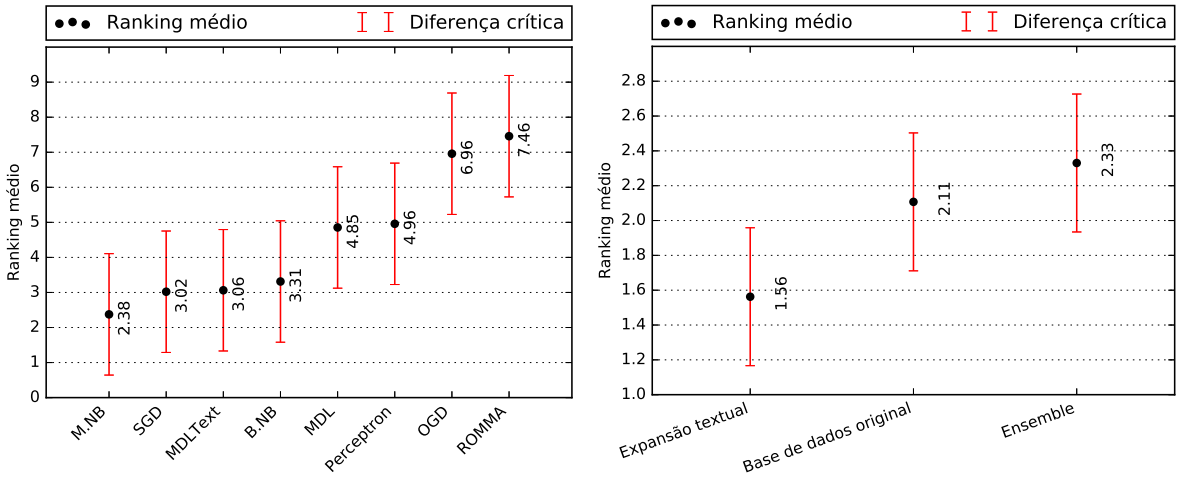
	Base de dados original	Expansão textual	Ensemble
B.NB	0,924	0,923	0,924
Perceptron	0,886	0,889	0,921
M.NB	0,906	0,911	0,909
MDL	0,897	0,901	0,900
`MDLText`	0,899	0,899	0,879
ROMMA	0,856	0,869	0,811
OGD	0,807	0,856	0,846
YSC - Psy			
`MDLText`	0,939	**0,948**	0,896
SGD	0,911	0,914	0,946
M.NB	0,941	0,936	0,944
B.NB	0,940	0,931	0,938
Perceptron	0,896	0,909	0,939
MDL	0,897	0,914	0,872
OGD	0,905	0,906	0,867
ROMMA	0,862	0,896	0,834
YSC - Shakira			
SGD	0,901	0,903	**0,923**
`MDLText`	0,911	0,920	0,888
Perceptron	0,896	0,891	0,912
M.NB	0,909	0,911	0,912
MDL	0,898	0,903	0,874
ROMMA	0,891	0,883	0,839
OGD	0,885	0,872	0,844
B.NB	0,875	0,882	0,869

É importante destacar que, para cada método de classificação e base de dados, a coluna `Expansão textual` da Tabela 21 reporta apenas o resultado com a regra de expansão em que a melhor macro F-medida foi obtida. Conforme recomendado por Almeida *et al.* (2016), neste trabalho optou-se por não apresentar os resultados de todas as dez bases de dados expandidas porque nenhuma regra de expansão obteve o melhor resultado para todos os métodos e bases de dados. Segundo Almeida *et al.* (2016), as regras de expansão são estatisticamente equivalentes e, portanto, a escolha da melhor regra para cada método de classificação pode ser feita usando-se uma busca em grade.

O `MDLText` obteve a melhor macro F-medida nos experimentos com a base de dados `YSC - Psy` e foi um dos quatro melhores métodos na maioria das bases de dados avaliadas. Individualmente, usando-se as mensagens de texto originais, o `MDLText` obteve a melhor macro F-medida na base de dados `YSC - Shakira`. Nos experimentos

com expansão, ele obteve a melhor macro F-medida na base de dados `YSC - Eminem`. No entanto, nos experimentos com a técnica *ensemble*, ele não obteve o melhor resultado em nenhuma base de dados.

Para certificar que os resultados não foram obtidos por acaso, foi realizada uma análise estatística usando o teste de Friedman. A Figura 30a mostra o *ranking* médio dos métodos baseado na macro F-medida obtida por eles. A Figura 18b apresenta o ranking médio obtido nos experimentos usando os documentos de texto originais, com expansão e com a técnica *ensemble*.



(a) (b)

Figura 30 – *Rankings* médios usados na análise estatística e diferenças críticas calculadas usando o teste *post-hoc* de Bonferroni–Dunn. A Figura 30a mostra o *ranking* médio para cada método. A Figura 30b mostra o *ranking* médio obtido nos experimentos usando os documentos de texto originais, a expansão e a técnica *ensemble*.

Para um intervalo de confiança $\alpha = 0{,}05$, o teste de Friedman indicou que a hipótese nula de igualdade entre os métodos pode ser seguramente descartada. Portanto, foi realizada uma comparação par-a-par usando o teste *post-hoc* de Bonferroni–Dunn.

Para um intervalo de confiança $\alpha = 0{,}05$, a diferença crítica calculada pelo teste *post-hoc* de Bonferroni–Dunn foi igual a 1,732. Portanto, há evidência estatística suficiente para afirmar que o desempenho geral do `MDLText` foi significativamente melhor que o desempenho geral dos métodos MDL, perceptron, OGD e ROMMA. Porém, ele foi estatisticamente equivalente aos métodos M.NB, SGD e B.NB.

Comparando-se os resultados obtidos com as bases de dados originais, as bases de dados expandidas e a técnica *ensemble*, a Tabela 21 mostra que, na maioria das bases de dados, o melhor resultado foi obtido nos experimentos com a técnica *ensemble*. Porém, em geral, o desempenho dos métodos de aprendizado *online* foram melhores usando a expansão textual. Isto pode ser confirmado pelos *rankings* médios mostrados na Figura

30b. Para certificar-se que tais diferenças foram significativas, também foi realizada uma análise estatística.

De acordo com o método não-paramétrico de Friedman, para um intervalo de confiança $\alpha = 0{,}05$, a hipótese nula de igualdade entre os resultados obtidos usando as bases originais, a expansão e a técnica *ensemble* pode ser descartada. De acordo com a comparação par-a-par usando o teste *post-hoc* de Bonferroni–Dunn, para um intervalo de confiança $\alpha = 0{,}05$, a diferença crítica entre os métodos foi 0,396. Portanto, pode-se afirmar que o desempenho obtido usando os documentos de texto após a aplicação da expansão textual foi estatisticamente superior ao desempenho obtido usando os documentos de texto originais e a técnica *ensemble*. Estes resultados estão em consonância com o que foi apresentado por Almeida *et al.* (2016).

A.4 Considerações finais

Neste apêndice, foi apresentado o estudo do desempenho do `MDLText` na classificação de documentos de texto curtos e ruidosos, com o auxílio de técnicas de normalização léxica e indexação semântica. Para isso, foram feitos experimentos com oito bases de dados reais e públicas de diferentes mídias. Os resultados obtidos pelo método proposto foram comparados com os resultados de métodos tradicionais de aprendizado *online*. A análise estatística dos resultados mostrou que o `MDLText` foi superior aos métodos MDL, ROMMA, perceptron e OGD.

Também foi avaliado se a aplicação de normalização léxica e indexação semântica pode trazer benefícios à classificação de documentos de texto curtos e ruidosos. Neste caso, a análise estatística mostrou que os métodos de classificação tiveram melhor desempenho quando tais técnicas foram aplicadas.

Por fim, foi proposta e avaliada uma técnica *ensemble* que combina as predições obtidas pelos métodos de classificação usando os documentos de texto originais e expandidos. Os resultados obtidos foram os melhores na maioria das bases de dados, porém estatisticamente inferior aos resultados individuais obtidos pelos métodos de classificação após a aplicação das técnicas de expansão.