5
Algoritmos-Base de Referência

O algoritmo BAS e seus derivados, como toda abordagem de Boosting, necessitam de um algoritmo-base de forma a realizar o seu treinamento iterativo.

Neste capítulo, apresentamos os algoritmos-base empregados pelas estratégias de Boosting utilizadas nos experimentos.

5.1
Toco de Decisão

Um dos algoritmos-base mais simples, utilizado em tarefas de aprendizado de máquina, é o Toco de Decisão (*Decision Stump*) (Fre96). O Toco de Decisão é um modelo simples que consiste em uma árvore de decisão com apenas um nível e duas ramificações. Normalmente, tal algoritmo não é utilizado sozinho em tais tarefas, pois é muito difícil atingir um bom desempenho utilizando apenas um atributo. Entretanto, ele é frequentemente utilizado em técnicas de comitê, incluindo Boosting, gerando múltiplos componentes para o algoritmo de aprendizado de comitê.

O *Toco de Decisão* implementado é o mesmo do pacote WEKA 3 (Wit02). Ele encontra a melhor divisão para atributos no conjunto de treinamento, maximizando a entropia dos atributos dos exemplos.

Exemplos de tocos de decisão, para atributos categóricos e numéricos, são mostrados na Figura 5.1.

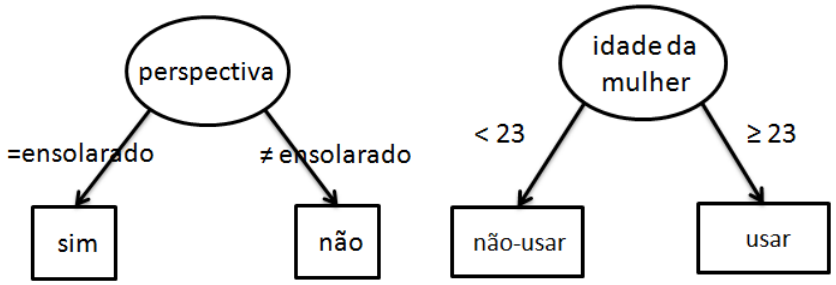

Figura 5.1: Exemplos de tocos de decisão para atributos categóricos e numéricos.

No exemplo da esquerda da Figura 5.1, extraído do conjunto *Weather*, o toco de decisão classifica todas as instâncias baseado no atributo categórico *perspectiva*. Caso ele seja igual a *ensolarado* classifica como *sim*, caso contrário, como *não*. Já no exemplo da direita da Figura 5.1, extraído do conjunto *Contraceptive Method Choice*, o toco de decisão classifica todas as instâncias baseado no atributo numérico *idade da mulher*. Caso ele seja menor que 23 classifica como *não-usar*, caso seja maior ou igual a 23 classifica como *usar*.

5.2 Aprendizado Baseado em Transformações

Aprendizado Baseado em Transformações (*Transformation-based Learning* - TBL) (Bri92) é um algoritmo de aprendizado de máquina que utiliza uma estratégia gulosa para correção de erros. O seu objetivo principal é criar uma lista ordenada de regras que corrige erros de classificação gerados por um classificador-base em um conjunto de treinamento. Tais regras são geradas a partir de gabaritos previamente determinados.

O classificador-base utilizado pelo TBL pode ser qualquer tipo de algoritmo. Entretanto, de forma a facilitar a modelagem TBL, normalmente é utilizado um algoritmo simples baseado em estatísticas de contagem do conjunto de treinamento.

Com esse objetivo, as entradas do algoritmo TBL são um conjunto de treinamento, um conjunto de gabaritos, um classificador base e um limiar de pontuação.

O aprendizado do algoritmo é um procedimento guloso dirigido pelos erros que iterativamente cria um conjunto de regras de transformação derivadas de um conjunto de gabaritos, maximizando a sua pontuação. A pontuação de uma regra em uma iteração do algoritmo é definida como sendo o número de correções que ela realiza no conjunto de treinamento, descontando o número de erros que ela realiza no mesmo conjunto. Em cada iteração, a regra com maior pontuação, melhor que o limiar, é escolhida para ser utilizada no classificador gerado. O limiar de pontuação pode ser ajustado para evitar um sobre-ajuste ao conjunto de treinamento.

A classificação de um novo dado é feita com a aplicação do classificador-base e, em seguida, do conjunto de regras ordenadas. O pseudo-código do algoritmo de aprendizado TBL é mostrado no Algoritmo 5.1 a seguir.

Gabaritos TBL

Um gabarito TBL pode ser qualquer combinação de atributos, posições e valores. Tal combinação deve ser relevante para a tarefa em questão. O gabarito

Algoritmo 5.1 Aprendizado TBL.

Entrada: conjunto de treinamento: Tr_0
conjunto de gabaritos: G
classificador-base: CB
limiar de pontuação: τ
$Tr_1 = CB(Tr_0)$ // Aplique o classificador-base CB no conjunto Tr_0
$R \leftarrow \{\}$ // Inicialize o conjunto de regras
$k \leftarrow 1$
repita
  RC_k = RegrasCandidatas(G, Tr_k) // Encontre todas as regras candidatas RC_k utilizando Tr_k e G
  para todo $r \in RC_k$ **faça**
    P(r) = #Correções(r) - #Erros(r) // Calcule a pontuação de cada regra candidata
  fim para
  r = Max(RC_k, P) // Escolha a regra com r maior pontuação
  se $P(r) > \tau$ **então**
    $Tr_{k+1} = r(Tr_k)$ // Aplique a regra r no conjunto Tr_k
    $R = R \cup \{r\}$. // Adicione a regra selecionada no conjunto de regras final
  fim se
  $k = k + 1$
até $P(r) \leq \tau$
Saída: $CB + R$

TBL é formalmente definido como uma sequência de *Termos Atômicos* (TAs). Um TA é a menor unidade de gabarito que indica o atributo e as condições de instanciação de um gabarito. Ele deve indicar o pedaço de contexto que uma regra TBL precisa testar quando aplicada no conjunto de treinamento. Alguns exemplos de termos atômicos são enumerados a seguir.

1. **atrib**[d], que verifica o atributo de um exemplo localizado d exemplos posteriores ou anteriores, dependendo do sinal, em relação ao exemplo atual. Como exemplo ilustrativo temos pos[-1].

2. **atrib**[d_1, d_2], que verifica o atributo de exemplos localizados no intervalo de exemplos posicionados entre d_1 e d_2, em relação ao exemplo atual. Como exemplo ilustrativo temos palavra[-2, 1].

3. **atrib**$[d_1, d_2]$_onde$\{\textbf{atrib}_2 = \text{v}\}$, que checa o atributo de exemplos localizados no intervalo de exemplos posicionados entre d_1 e d_2, em relação ao exemplo atual e que possuem o atributo $atrib_2$ igual a v. Como exemplo ilustrativo temos palavra$[-3, 2]$_onde$\{\text{pos} = \text{NNP}\}$.

Termos atômicos mais complexos podem ser definidos para criar regras mais especializadas.

5.3 TBL Genético

Nesta seção, apresentamos a codificação genética desenvolvida para a determinação automática dos gabaritos TBL. A utilização de algoritmos genéticos em conjunto com o algoritmo TBL foi explorada (Wil05) no processo de geração das regras instanciadas com o objetivo de prover uma ordenação adaptativa. Entretanto, tal abordagem nunca foi utilizada com o objetivo de gerar gabaritos automáticos.

A abordagem genética TBL-GA (Mil07a, Mil07b) utiliza um *cromossomo* que representa uma sequência de termos atômicos possíveis de serem combinados em um gabarito. Cada valor de cromossomo indica o índice do TA correspondente em uma lista fixa fornecida previamente. Cada gabarito é representado por uma porção do *cromossomo* de tamanho fixo que estabelece um tamanho máximo para todos os gabaritos. O valor -1 é utilizado para indicar a ausência de um TA em uma determinada posição. A repetição de TAs em um mesmo gabarito é uma possibilidade, porém tais TAs são simplesmente descartados.

Tabela 5.1: Exemplo de uma lista de possíveis TAS para a codificação genética.

Índice na Lista	0	1	2	3	4	5
Termo Atômico	$a[-1]$	$a[-2]$	$b[0]$	$b[1]$	$a[0,2]$	$b[-2,-0]$

Os parâmetros de entrada necessários para realizar o treinamento de tal codificação são a lista de possíveis TAs, o número de gabaritos a serem gerados e o tamanho máximo dos gabaritos. Um exemplo de tal codificação é mostrado na Tabela 5.2. Tal exemplo mostra a codificação de três gabaritos de tamanho máximo 4 utilizando os seis possíveis TAs mostrados na Tabela 5.1.

Tabela 5.2: Exemplo de uma codificação genética para o TBL-GA.

	Gabarito 1				Gabarito 2				Gabarito 3			
	TA_1	TA_2	TA_3	TA_4	TA_1	TA_2	TA_3	TA_4	TA_1	TA_2	TA_3	TA_4
C_r	1	3	-1	0	5	1	3	2	1	2	1	3

O *cromossomo* apresentado na Tabela 5.2 gera os gabaritos mostrados na Tabela 5.3.

O treinamento dos *cromossomos* é realizado da seguinte maneira. Inicialmente, um extrato do conjunto de treinamento é escolhido para a abordagem

Tabela 5.3: Exemplo de gabaritos gerados para o TBL-GA.

	Índices				Termos Atômicos				Gabarito
1	1	3	-1	0	$a[-2]$	$b[1]$	$\oslash$	$a[-1]$	$a[-1]$ $a[-2]$ $b[1]$
2	5	1	3	2	$b[-2,-0]$	$a[-2]$	$b[1]$	$b[0]$	$a[-2]$ $b[0]$ $b[1]$ $b[-2,-0]$
3	1	2	1	3	$a[-2]$	$b[0]$	$a[-2]$	$b[1]$	$a[-2]$ $b[0]$ $b[1]$

genética. Tal extrato é divido em dois subconjuntos, o de treinamento genético e o de validação, que são utilizados, respectivamente, para treinar e avaliar o desempenho dos indivíduos.

Cada indivíduo é um classificador TBL treinado no conjunto de treinamento genético e aplicado no conjunto de validação. A função de aptidão do indivíduo é dada pelo valor do desempenho alcançado por ele no conjunto de validação.

Os operadores de cruzamento e mutação utilizam abordagens genéticas tradicionais. O operador de cruzamento gera um novo indivíduo combinando dois *cromossomos* quebrados em um ponto aleatório. Já o operador de mutação gera um novo indivíduo realizando uma alteração no *cromossomo* que implica em uma troca do TA utilizado por um gabarito aleatório.

O **melhor** indivíduo encontrado pelo algoritmo genético é então treinado no conjunto de treinamento completo.

5.4 Aprendizado de Transformação guiado pela Entropia

Aprendizado de Transformação guiado pela Entropia (Entropy-guided Transformation Learning - ETL) (Mil08a) é uma estratégia de aprendizado de máquina que combina as vantagens das árvores de decisão e do TBL. A ideia principal do ETL é utilizar a indução das árvores de decisão para obter gabaritos. Após isso, o algoritmo TBL é utilizado para gerar regras de transformação. Esse método é ilustrado na Figura 5.2

Existem diversas possibilidades para extrair combinações de atributos de árvores de decisão. Em um caminho da raiz até as folhas, atributos mais informativos aparecem primeiro. Como o objetivo é gerar os gabaritos mais promissores, somente é necessário combinar tais atributos.

O processo envolvido na extração dos gabaritos da árvore de decisão inclui um caminhamento em profundidade na árvore. Para cada nó visitado, um novo gabarito é criado combinando o gabarito gerado pelo nó predecessor com o atributo utilizado para dividir os dados no nó. Apesar de ser um esquema bem simples, os seus resultados são extremamente eficientes.

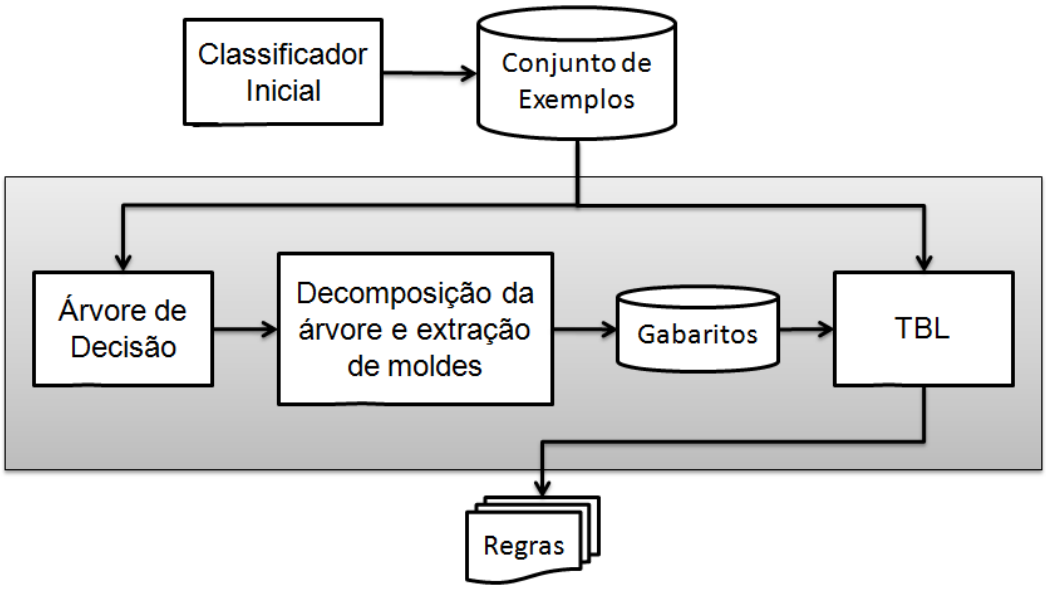

Figura 5.2: Aprendizado de transformação guiado pela entropia.

A Figura 5.3 mostra o exemplo de uma possível árvore gerada para o conjunto de dados *Weather*, descrito na Subseção 6.3.1. Utilizando o método descrito para extrair gabaritos de tal árvore, a lista de gabaritos mostrada no lado esquerdo da Tabela 5.4 é obtida. De forma a gerar mais combinações de atributos, sem aumentar em muito o número de gabaritos, a lista de gabaritos é estendida, incluindo gabaritos que não possuem o atributo do nó raiz. Tal extensão na lista é exemplificada no lado direito da Tabela 5.4.

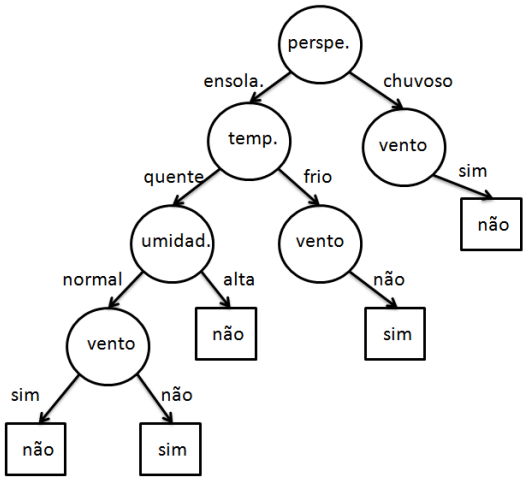

Figura 5.3: Exemplo de árvore de decisão para a tarefa *Weather*.

Tabela 5.4: Exemplo de lista de gabaritos para a tarefa *Weather*.

Lista de Gabaritos	**Extensão da Lista de Gabaritos**
perspectiva	
perspectiva temperatura	temperatura
perspectiva temperatura umidade	temperatura umidade
perspectiva temperatura umidade vento	temperatura umidade vento
perspectiva temperatura vento	temperatura vento
perspectiva vento	vento

5.4.1 Extensões ao ETL

O algoritmo ETL original possui um bom desempenho para diversas tarefas de classificação. Entretanto, com o objetivo de melhorar ainda mais o seu desempenho e gerar uma maior variabilidade quando utilizando em estratégias de comitê, algumas extensões ao algoritmo (San09) podem ser empregadas.

Regras redundantes

Normalmente o TBL aprende apenas uma regra por iteração. Entretanto, mais de uma regra candidata pode possuir a mesma pontuação. Nesse caso, um dos diversos critérios de desempate é empregado na escolha da regra vencedora.

Quando regras redundantes são geradas, em cada iteração, além da melhor regra, o algoritmo aprende todas as regras que não provocam erros e que corrigem exatamente os mesmos exemplos que a regra vencedora. Tais regras não alteram o treinamento do TBL pois não provocam nenhum erro no conjunto de treinamento.

Tal extensão tem um papel chave quando da utilização de estratégias derivadas do algoritmo TBL em uma abordagem de comitê, pois permitem ao algoritmo TBL gerar a diversidade necessária para que os classificadores gerados formem um comitê com desempenho melhor que o desempenho de apenas um classificador TBL.

Atributos com alta dimensionalidade

Atributos com alta dimensionalidade são caracterizados por possuírem uma grande quantidade de valores possíveis. Em tarefas de recuperação de informação, por exemplo, o atributo palavra pode assumir milhares de valores.

Comumente, as árvores de decisão empregadas na obtenção dos gabaritos, que são guiadas pelo ganho de informação do atributo, não utilizam esses atributos.

Com o objetivo de contornar esse problema, o ETL pre-processa o conjunto de dados, mantendo apenas os valores mais informativos. Tal processo é conduzido calculando-se o ganho de informação por valor possível do atributo. Tais valores são então ordenados por esse ganho e os que possuem um baixo ganho de informação individual são substituídos por um valor postiço.

Subamostragem de gabaritos

As árvores de decisão utilizadas pelo algoritmo ETL normalmente geram uma grande quantidade de gabaritos. Tal fato pode inviabilizar o treinamento do algoritmo TBL, principalmente quando utilizado dentro de uma estratégia de comitê. Nesse caso, com o objetivo de diminuir o tempo de treinamento e dar uma maior variabilidade aos múltiplos classificadores gerados no comitê, é empregada a subamostragem aleatória de gabaritos, onde para cada classificador TBL, somente uma fração dos gabaritos gerados é utilizada.

Subamostragem de atributos

Também com o objetivo de diversificar os classificadores gerados em um comitê e economizar tempo de treinamento pode ser utilizada uma técnica comum de geração de múltiplo classificadores que é a subamostragem aleatória de atributos. Essa estratégia é bem interessante para tarefas com grandes quantidades de atributos.