3
Extração de Regras Simbólicas a partir de Máquinas de Vetor Suporte

3.1.
Introdução

Como já mencionado na seção 1.1, as SVMs geram, da mesma forma que redes neurais (RN), um "modelo caixa preta" de difícil interpretação sobre a relação entre as variáveis de entrada e a variável de saída.

No caso de redes neurais, vários métodos já foram desenvolvidos, traduzindo o conhecimento adquirido pelas redes neurais treinadas para uma representação interpretável [44], [45], [46], [47], [48], [49].

No caso de SVM, por ser um tópico de pesquisa mais recente, esse aspecto ainda não foi extensamente desenvolvido. Entretanto, nota-se um interesse crescente nessa linha de pesquisa [50]. Duas propostas expostas em [13], [14] e [15] podem ser destacadas.

As seções seguintes descrevem os fundamentos básicos dos métodos RulExSVM [13] e SVM+Prototypes [14], [15].

3.2. Algoritmo RulExSVM

Faz-se, a seguir, uma descrição do algoritmo RulExSVM, conforme apresentado em [13].

Como foi visto na seção 2, os vetores suporte gerados através da otimização do problema dual localizam-se próximo à fronteira de decisão. Se os pontos que não são vetores suporte fossem removidos do conjunto de treinamento, o mesmo hiperplano de separação seria obtido. Como os vetores suporte definem o hiperplano de separação, as regras associadas à fronteira de um hiper-retângulo – chamadas regras hiper-retangulares – são geradas diretamente a partir desses vetores.

O algoritmo de extração de regras RulExSVM explora o fato de que decisões de uma SVM não linear podem ser decodificadas em regras baseadas em informações provenientes de vetores suporte e de sua função de decisão. Esse método consiste em três fases: a fase inicial, a fase de ajuste e a fase final.

a) Fase inicial de geração de regras

Todos os atributos de entrada são normalizados entre [0,1].

Notação utilizada:

n: a dimensão dos dados;

$A1$: um vetor suporte da classe 1;

$A2$: um vetor suporte da classe 2;

N_1: número de vetores suporte da classe 1;

N_2: número de vetores suporte da classe 2;

$N_s = N_1 + N_2$: número total de vetores suporte;

$s_m = (s_{m1}, s_{m2},..., s_{mn})$: m-ésimo vetor suporte da classe 1;

$x = (x_1, x_2, ..., x_n)$: um ponto do espaço de entrada.

Para que fique mais claro o procedimento detalhado no algoritmo RulExSVM, considera-se o exemplo bidimensional exposto na Figura 5(a) extraído de [13], onde os vetores suporte da classe 1 são os pontos pretos A, B, C, D, E e F e os da classe 2, os pontos brancos G, I e J. À medida em que se descreve o algoritmo, faz-se em paralelo a ligação com o exemplo citado.

A regra hiper-retangular derivada do vetor suporte s_m da classe1 pode ser representada por:

$$\{ s_{mi} + l_{2i} \geq x_i \geq s_{mi} - l_{1i} , i = 1, ...,n\}, \text{ onde } 1 \geq l_{pi} \geq 0, p = \{1,2\}.$$

Sejam $L_o(i) = s_{mi} - l_{1i}$ e $H_o(i) = s_{mi} + l_{2i}$, respectivamente, os limites inferior e superior da regra hiper-retangular ao longo da *i*-ésima dimensão. Esses limites são determinados, inicialmente, pela função de decisão *f* que distingue a classe 1 da classe 2, como descrito abaixo.

Pode ser visto na Figura 5(a) que, para cada um dos eixos, é traçada uma reta paralela aos mesmos, a partir de um vetor suporte da classe 1. Essa reta pode ser estendida nas duas direções. Através da geração dessas retas pode-se determinar os pontos de interseção entre essa linha e a fronteira de decisão, como é indicado no algoritmo a seguir.

Dada uma SVM treinada, a regra baseada no vetor suporte s_m pode ser gerada pelo seguinte algoritmo:

1. faça $d = 1$, *d* se refere a dimensão;
2. calcule x_d sujeito a $f(x) = 0$ e $x_j = s_{mj}$ ($j = 1,...,n$ e $j \neq d$) pelo método de Newton [51], onde *f* é a função de decisão da SVM;
3. determine L_o e H_o de acordo com a solução do problema no passo 2. O número de soluções para o problema do passo 2 pode ser diferente:
   i. se não existe solução, isto é, não existe interseção entre a linha estendida a partir de s_m na dimensão d e a fronteira de decisão, então $L_o(d) = 0$ e $H_o(d) = 1$;
   ii. se existe uma solução
      se $s_{mj} \geq x_d$, então $L_o(d) = x_d$ e $H_o(d) = 1$,
      caso contrário $L_o(d) = 0$ e $H_o(d) = x_d$;
   iii. se existem duas soluções, x_{d1} e x_{d2} ($x_{d1} \leq x_{d2}$), então $L_o(d) = x_{d1}$ e $H_o(d) = x_{d2}$;
   iv. se existem mais de duas soluções, as soluções x_{d1} e x_{d2} que estão mais próximos de s_{mj} são escolhidas com a condição de que $x_{d1} \leq s_{mj}$, $x_{d2} \geq s_{mj}$ e o conjunto de pontos $\{x | x_j = s_{mj}, j = 1,..., n, j \neq d, x_{d1} \leq x_d \leq x_{d2}\}$ são da mesma classe que o vetor suporte s_m, então $L_o(d) = x_{d1}$ e $H_o(\mathrm{d}) = x_{d2}$;
4. $d = d + 1$; se $d < n$, vá para o passo 2, se não, fim.

No exemplo em questão, o vetor suporte A (da classe 1), os pontos de interseção entre as linha estendidas e a fronteira de decisão são mostradas na Figura 5(a). A fronteira inicial do hiper-retângulo, mostrada na Figura 5(b) com linha pontilhada, pode ser obtida através desses pontos de interseção. Como há

somente um ponto de interseção no eixo vertical, que define o limite inferior (caso ii do algoritmo acima), o limite superior será 1 (todos os atributos dos pontos de entrada estão entre 0 e 1).

b) Fase de ajuste das regras iniciais

A fase de ajuste do método RulExSVM é realizada após uma regra inicial ser gerada como mostrado na seção anterior. Esse ajuste é feito de modo que qualquer ponto da classe 2 seja excluído da região da regra.

O número de pontos cobertos por uma regra está relacionado ao volume da região associada a essa regra. Assim, o objetivo dessa fase é excluir os pontos da classe 2 de modo a se obter um hiper-retângulo com o maior volume possível. Ao se considerar um espaço de dimensão *n*, existem *n* possibilidades de se excluir um ponto, cada uma delas em uma das suas *n* dimensões. A escolha da dimensão em que o ponto não pertencerá à regra é feita de modo que o volume restante seja o maior possível. Isso é feito da seguinte forma:

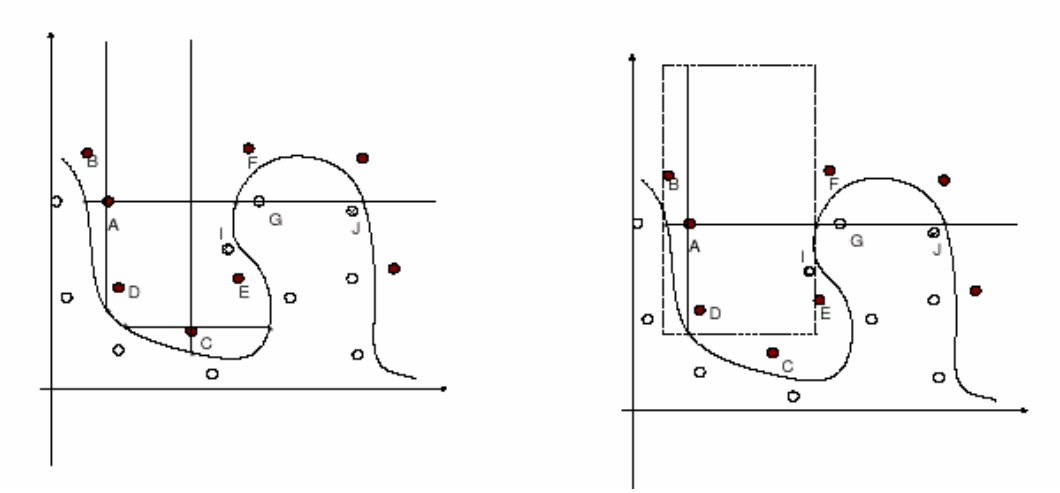
- para uma dada regra, ache todos os pontos da classe 2 que pertencem a essa regra e chame esse conjunto de Q;
- escolha aleatoriamente um dos pontos e o chame de P;
- calcule as distâncias de P às fronteiras do hiper-retângulo em cada dimensão;
- retire P do hiper-retângulo, diminuindo a regra hiper-retangular no eixo que mantiver o máximo volume do hiper-retângulo;
- escolha outro ponto de Q e repita o procedimento até todos os pontos de Q terem sido analisados.

c) Fase final

Regras de classes diferentes podem se interceptar. Se o hiper-retângulo associado a uma regra estiver totalmente contido no hiper-retângulo associado a uma outra regra, a primeira regra será descartada. Essa fase elimina as regras redundantes.

A seguir é explicado o procedimento para a retirada das regras redundantes:

- ache os padrões que satisfazem a cada regra;
- se o conjunto de pontos que satisfazem a uma regra r está contido em algum conjunto de pontos que satisfazem a uma outra regra s, então a regra r é removida.



(a) (b)

Figura 5 - RulExSVM

3.3.
Algoritmo SVM+Prototypes

Uma outra forma de se construir um extrator de regras é através da definição de elipsóides no espaço de entradas. A maneira como o algoritmo de extração de regras SVM+Prototypes [14], [15] define esses elipsóides é apresentada a seguir.

O método também utiliza a informação dada pelos vetores suporte, usados para determinar as fronteiras das regiões definidas no espaço de entrada. Essas regiões são obtidas pela combinação de vetores protótipos (centros de cada classe) e de vetores suporte. Os vetores protótipos são computados através de um algoritmo de clusterização. Cada região define uma regra com uma sintaxe: regras com equações, que correspondem a equações matemáticas de elipsóides, e regras com intervalos, associadas a hiper-retângulos definidos por elipsóides paralelos aos eixos coordenados. A Figura 6, extraída de [15], mostra exemplos desses tipos de regras. Na Figura 7, são mostradas duas partições, P1 e P2, com centros c1 e c2, onde os vetores suporte estão destacados.

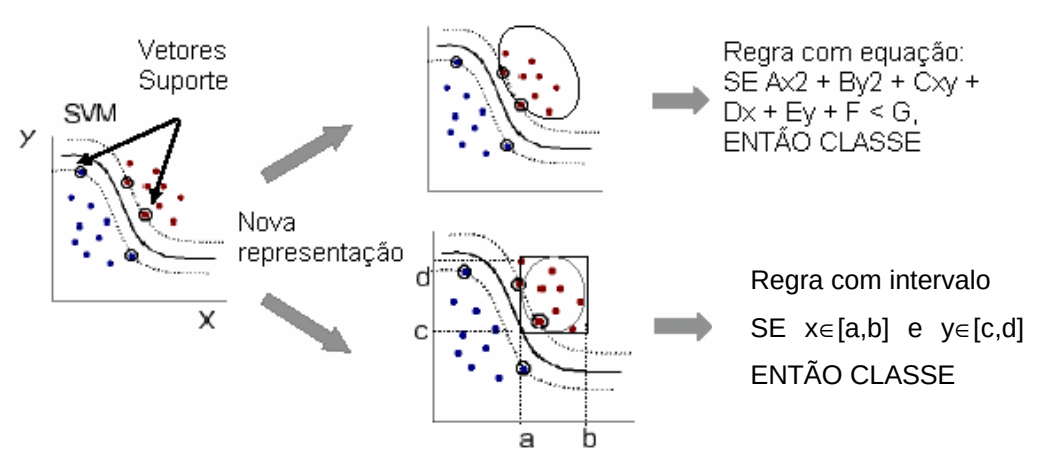


Figura 6 - Regras geradas pelo algoritmo SVM+Prototypes

Define-se um elipsóide pelo protótipo, que é o seu centro, e pelo vetor suporte pertencente à partição. Na construção do método, o vetor suporte escolhido é o mais distante do protótipo. A linha reta definida por esses dois pontos é o primeiro eixo do elipsóide. Na Figura 7, para a partição 1, o vetor suporte v1 é o mais distante de c1 e é o escolhido pelo método para definir o primeiro eixo do elipsóide. Por simples geometria, determinam-se os outros eixos e os vértices correspondentes. Há duas possibilidades para a definição

desses vértices: calculá-los a partir dos vetores suporte ou usar o ponto mais afastado do protótipo. As Figuras 8 e 9 mostram a construção do segundo eixo do elipsóide e o elipsóide para cada um dos casos acima para a partição P1, onde v2 é o vetor suporte usado para definir o 2º eixo do elipsóide (Figura 8) e a1 é o ponto mais afastado do protótipo c1 (Figura 9).

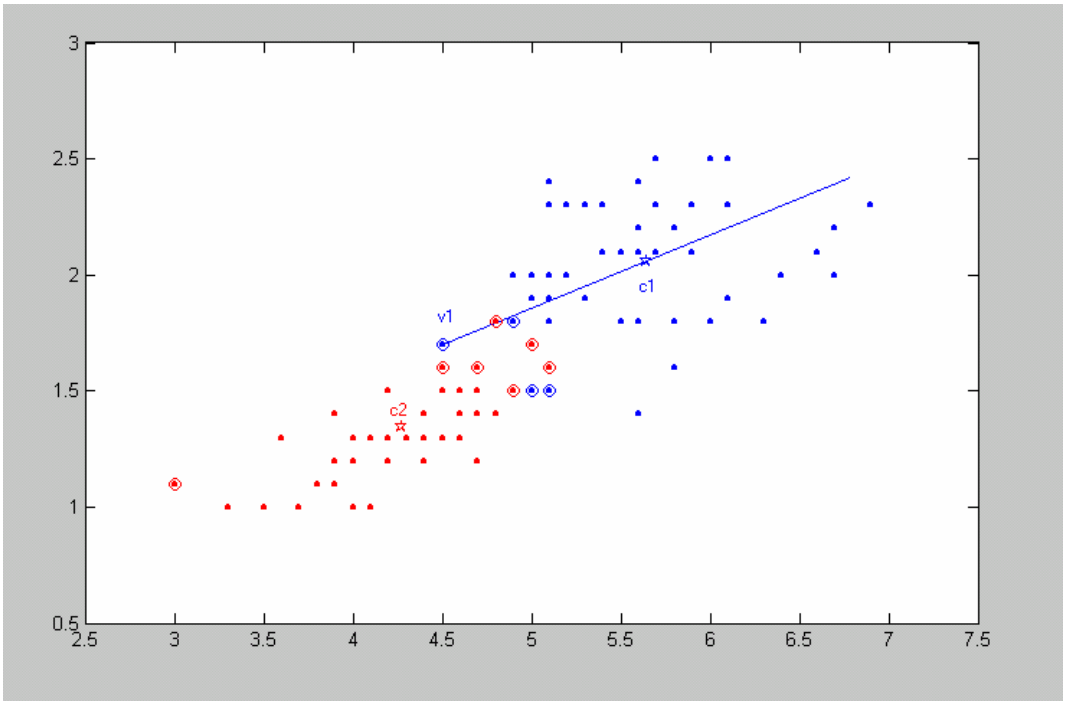


Figura 7 - Partições e 1º eixo do elipsóide

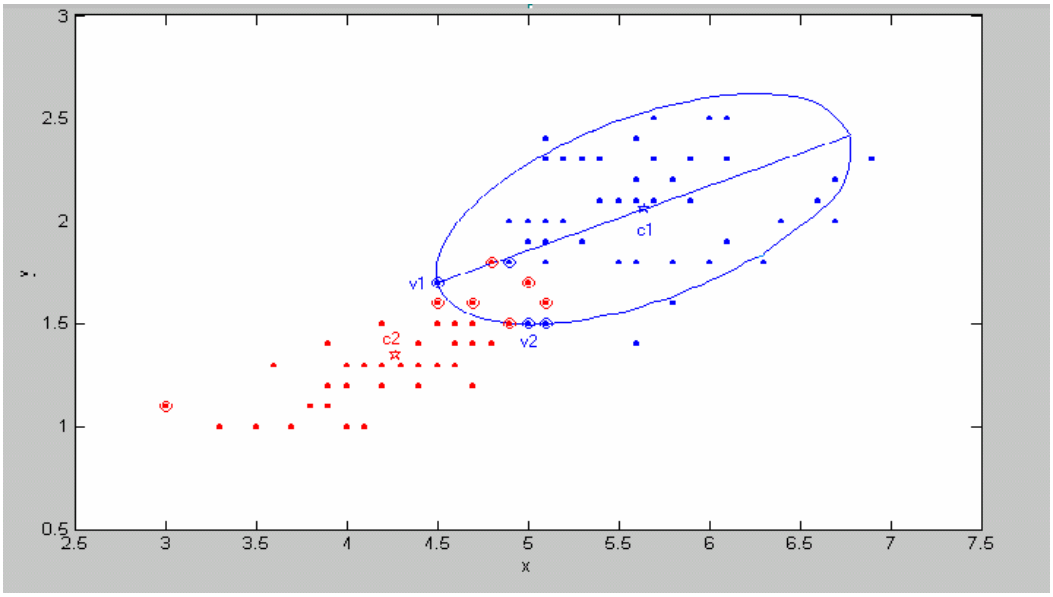


Figura 8 - Construção do 2º eixo do elipsóide usando o vetor suporte v2

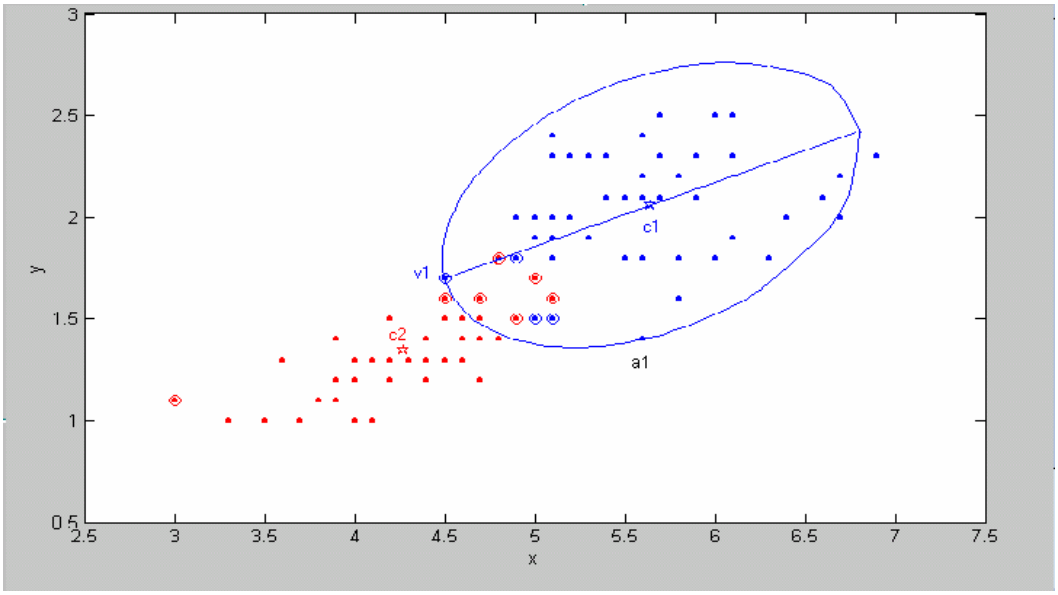


Figura 9 - Construção do 2º eixo do elipsóide usando o ponto mais distante do protótipo

A construção de hiper-retângulos é similar, com a diferença de que linhas paralelas aos eixos coordenados, e não aos eixos do elipsóide, são usadas para definir os eixos da região associada. Na Figura 10, é mostrado o retângulo construído para a partição 1, usando os vetores suporte v1 e v2.

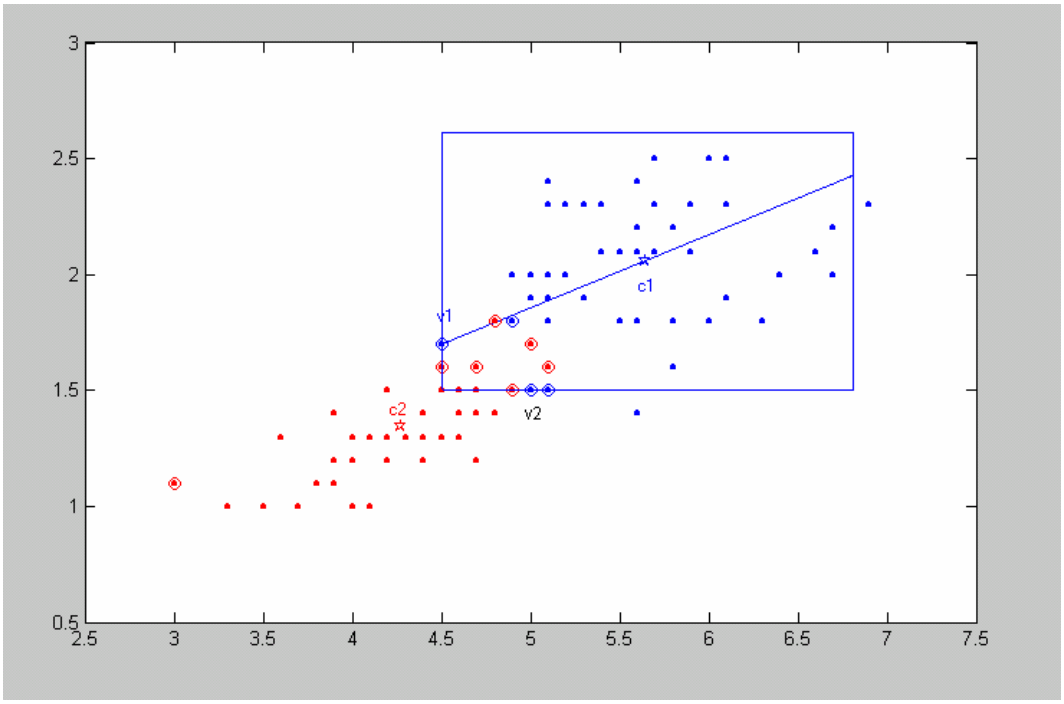


Figura 10 - Construção do hiper-retângulo

Para se determinar o número de elipsóides por classe, o algoritmo segue um esquema incremental. Para cada classe, começando com um único

protótipo, gera-se o elipsóide associado. Em seguida, é feito um teste de partição para verificar a existência de vetores suporte de outras classes no elipsóide. Se não houver, a região é transformada em uma regra. Caso contrário, a região é dividida e novas regiões são geradas. O procedimento com o teste de partição e divisão do elipsóide é repetido enquanto houver uma região com vetores suporte de outras classes ou até que o número máximo de iterações seja atingido. Esse processo controla o número de regras geradas. A Figura 11, extraída de [15], mostra em (a) um elipsóide gerado pelo procedimento acima descrito. Pode-se ver que o elipsóide contém vetores suporte de outras classes, que estão acima da fronteira de decisão. Dividindo-o (Figura 11(b)), os dois elipsóides resultantes não contêm vetores suporte de outras classes e representam melhor a classe.

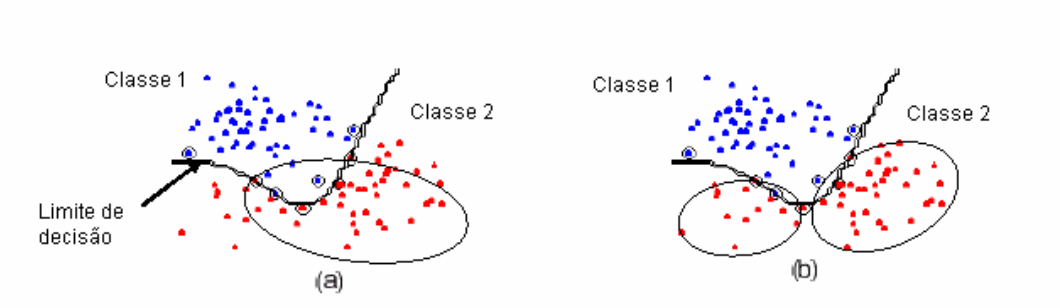


Figura 11 - Geração e divisão de um elipsóide
(a) Geração de um elipsóide. (b) O elipsóide de (a) dividido em dois elipsóides

3.4. Comentário sobre as deficiências dos métodos

Como já mencionado, ambos métodos geram regras cujos antecedentes são intervalos ou equações. Essa característica torna as regras menos interpretáveis e prejudica a extração de conhecimento útil.

Como já mencionado, o método RulExSVM utiliza o hiperplano de decisão para definir os hiper-retângulos. No caso de classificação em mais de duas classes, em geral, não existe um hiperplano de decisão, o que torna esse método difícil de ser estendido para mais de duas classes. Quanto ao método SVM + Prototypes, a extração de regras depende muito da escolha do algoritmo de clusterização tanto em número de regras quanto na acurácia das regras. De acordo com [14] e [15], a partição do cluster só depende dos protótipos e dos vetores suporte, o que pode levar a uma acurácia baixa das regras, pois pontos de outras classes que não são vetores suporte não podem ser detectados e retirados da região da regra pelo método. Além disso, a determinação das fronteiras das regras torna-se complicada com o aumento da dimensionalidade dos dados porque não pode ser resolvido por simples geometria.