DOI: 10.48011/asba.v2i1.1726

Aprendizado Ativo via Algoritmo de Evolução Diferencial

Marcus Vinicius de Freitas Diadelmo [*]
Marcus Vinicius de Paula [**] **Tamires Martins Rezende** [**]
Antônio de Pádua Braga [**] **Cristiano Leite de Castro** [**]

[*] *Instituto Federal de Minas Gerais (IFMG)*
Rua José Benedito, 139 - 35450-000 - Santa Efigênia - Itabirito, Minas Gerais, Brasil (e-mail: marcus.diadelmo@ifmg.edu.br).
[**] *Programa de Pós-Graduação em Engenharia Elétrica - Universidade Federal de Minas Gerais (UFMG)*
Av. Antônio Carlos, 6627 - 31270-901 - Belo Horizonte, Minas Gerais, Brasil (e-mail: marcusdepaula@ufmg.br, inc.tamires@gmail.com, apbraga@ufmg.br, crislcastro@ufmg.br)

Abstract: This work presents a methodology for data classification, in two stages, using an active learning technique. The first step of the methodology is to determine a starting point for choosing the initial data to be labeled. In the second step, the active learning algorithm is applied to the data. Since that the labeling can be expensive in several applications, the objective of the proposed methodology is to obtain a good performance of the active learning algorithm with the least number of labeled data. The proposed method has a priori information on the distribution of unlabeled data. Using this distribution, the population of the differential evolution algorithm converges to the region of class separation. The population used in this task is distributed in the domain of unlabeled data and evolves to regions of low data density. At the end of the execution of the evolutionary algorithm, a hyperplane is generated using the final population. Then, begins the active learning process (pool-based). The classifier is obtained using a SVM (Support Vector Machine) algorithm. Through of the experimental results, we found that the active learning algorithm obtained an accuracy close to the maximum (when all data are used for the training) with less than 20% of the labeled data. We conclude that the priori knowledge of the distribution of classes allows to obtain accurate results considering a relatively small number of labeled data.

Resumo: Este trabalho apresenta uma metodologia para classificação de dados, em duas etapas, utilizando uma técnica de aprendizado ativo. A primeira etapa da metodologia consiste em determinar um ponto de partida para a escolha dos dados iniciais a serem rotulados. Na segunda etapa, o algoritmo de aprendizado ativo é aplicado aos dados. Uma vez que a rotulação pode apresentar alto custo em diversas aplicações, o objetivo da metodologia proposta consiste em obter uma boa performance do algoritmo de aprendizado ativo com o menor número possível de dados rotulados. O método proposto possui informações a priori sobre a distribuição dos dados não rotulados. Por meio dessa distribuição, a população do algoritmo de evolução diferencial converge para próximo da região de separação das classes. A população empregada nesta tarefa é distribuída no domínio dos dados não rotulados e evolui para regiões de baixa densidade de dados. Ao finalizar a execução do algoritmo evolucionário, um hiperplano é gerado usando a população final. Em seguida, o processo de aprendizado ativo ($pool - based$) é iniciado. O classificador é obtido utilizando um algoritmo de máquina de vetores de suporte (SVM). Por meio dos experimentos realizados, verificou-se que o algoritmo de aprendizado ativo obteve uma acurácia de acertos próximo da máxima (quando todos os dados são utilizados para o treinamento) com menos de 20% dos dados rotulados. Conclui-se que o conhecimento a priori da distribuição das classes permite obter resultados acurados considerando um número relativamente pequeno de dados rotulados.

Keywords: Machine Learning; Active Learning; Evolutionary Algorithm; Differential Evolution; Support Vectors Machine.

Palavras-chaves: Aprendizado de Máquina; Aprendizado Ativo; Algoritmo Evolucionário; Evolução Diferencial; Máquina de Vetores de Suporte.

1. INTRODUÇÃO

O aprendizado de máquina supervisionado tem como finalidade realizar uma previsão por meio de um conjunto conhecido de dados de entrada e saída, denominados dados rotulados (Vapnik, 1999). A tarefa de rotulação dos dados pode apresentar um custo baixo ou elevado dependendo de sua aplicação (Zhu, 2005). De acordo com Settles (2008), a classificação de *spam*, por exemplo, é considerada uma classificação sem custo ou com baixo custo, uma vez que o próprio usuário rotula seus *emails* como sendo ou não pertencentes à classe *spam*. O mesmo é observado, segundo Settles (2008), na classificação de filmes cinco estrelas em *sites*. Novos filmes serão indicados posteriormente ao usuário por meio de um filtro construído por ele mesmo.

Por outro lado, existem problemas em que a rotulação de dados é um processo custoso. De acordo com Zhu (2005), a tarefa de reconhecimento de voz é um exemplo clássico de rotulação de alto custo, pois necessita de um tempo elevado e de um linguista. Uma boa extração de informação também possui um custo de rotulação elevado, pois exige anotações detalhadas. Segundo Rocca et al. (2011), a grande dificuldade está na obtenção destas anotações, pois é necessário uma pesquisa aprofundada e detalhada das informações.

Para reduzir os custos com a rotulação de dados, pode ser utilizado, como alternativa, o algoritmo de aprendizado ativo (Storn and Price, 1997a). Este algoritmo visa obter um hiperplano separador, por meio do aprendizado supervisionado, sem a necessidade de rotular todos os dados de treinamento. De acordo com Settles (2008), o algoritmo de aprendizado ativo determina, durante as iterações, quais dados serão rotulados para o aprendizado supervisionado.

Considerando que a escolha dos dados rotulados pesam significativamente no processo de aprendizado ativo, este trabalho propõe a utilização de uma técnica de computação evolutiva, denominada Evolução Diferencial (DE) (Holland et al., 1992), para definir um ponto de partida para o algoritmo de aprendizado ativo e reduzir o número de dados rotulados.

O artigo encontra-se organizado da seguinte maneira: as Seções 2, 3 e 4 apresentam uma breve revisão sobre as técnicas utilizadas neste trabalho. Na Seção 5 é apresentada a metodologia desenvolvida no trabalho. A Seção 6, por sua vez, apresenta os resultados dos testes realizados para validação da proposta. A Seção 7, por fim, apresenta as conclusões do trabalho.

2. APRENDIZADO ATIVO

Devido às dificuldades de rotulação de dados em determinados problemas, Brinker (2003) apresenta uma metodologia conhecida como Aprendizado Ativo. Nessa técnica, a escolha de novos dados a serem rotulados é determinada através da classificação de incerteza de pertencimento a uma determinada classe. As amostras são selecionadas com base na sua capacidade de representar uma melhor informação para o processo de treinamento. Uma das estratégias que pode ser utilizada no aprendizado ativo é a aprendizado $pool - based$.

Neste tipo de aprendizado, o conjunto de treinamento possui dados rotulados e não rotulados (*pool*). Durante o treinamento, o sistema escolhe determinados dados para serem rotulados e auxiliar o aprendizado (Lewis and Ringuette, 1994). Ao determinar um dado a ser rotulado, o sistema recorre ao oráculo (sistema capaz de definir o rótulo de qualquer um dos dados). A Figura 1 apresenta um diagrama esquemático deste tipo de aprendizado.

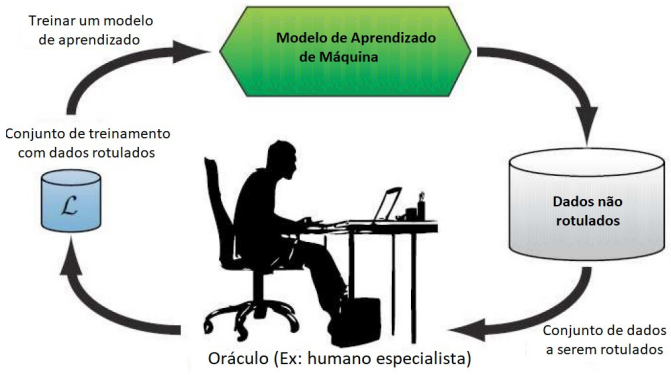


Figura 1. Aprendizado ativo $pool - based$. Adaptado de Settles (2008).

Por meio de um conjunto de dados rotulados é gerado um modelo. Em seguida, esse modelo é utilizado para determinar quais dados possuem uma melhor informação, ou seja, quais dados, no conjunto de dados não rotulados, tem maiores probabilidades de serem classificados de forma incorreta. Por fim, os dados selecionados são rotulados pelo oráculo. Um pseudo-código genérico dessa metodologia é apresentado no *Algorithm* 1.

Algorithm 1 Aprendizado Ativo

L - Conjunto de dados rotulados
U - conjunto de dados não rotulados (*pool*)
$\phi(\cdot)$ - Estratégia de consulta
B - Número de dados rotulados por treino
while not condição de parada **do**
  $\theta \longleftarrow train(L)$
  for b=1 to B **do**
    $x_b^* \longleftarrow argmax_{x \in U} \phi(x)$
    $y_b \longleftarrow rotulo(x_b^*)$
    $L \longleftarrow L \cup (x_b^*, y_b)$
    $U \longleftarrow U - x_b^*$
  end for
end while

3. ALGORITMO DE EVOLUÇÃO DIFERENCIAL

O Algoritmo de Evolução Diferencial é um algoritmo genético que utiliza operações de cruzamento (recombinação), mutação e seleção de indivíduos, inspirados na vida biológica, com objetivo de encontrar o valor ótimo de uma função de custo utilizando sucessivas gerações (Holland et al., 1992). O método DE utiliza um vetor de diferença para perturbar os indivíduos de uma população. Dessa forma a codificação real das variáveis é utilizada (Storn and Price, 1997b). O algoritmo utiliza um conjunto de soluções candidatas representadas na Equação 1:

$$X_{i,G} = \{x^1_{i,G}, \cdots, x^D_{i,G}\},\ i = 1, \cdots, NP, \qquad (1)$$

em que D representa o número de variáveis, NP é o número de indivíduos (soluções) da população (conjunto de soluções) e G é o número de gerações (iterações). Dessa forma, na Equação 1, $X_{i,G}$ representa o *i-ésimo* indivíduo, de dimensão D, na geração G. A população inicial é gerada de forma aleatória dentro dos limites mínimos e máximos das variáveis impostas pelo problema. Em seguida, aplicam-se a mutação e o cruzamento que são sucedidos pelo processo de seleção. Essas etapas são descritas nas Subseções 3.1 a 3.3 e exemplificadas pelo *Algorithm* 2.

Algorithm 2 Evolução Diferencial

```
t ← 1
X_t = {x_{t,i}; i = 1, 2, ··· , NP} - População Inicial
Avaliar X_t
while not condição de parada do
    for i = 1 to NP do
        Selecione aleatoriamente r_1, r_2 e r_3 ∈ 1, ··· , NP
        Selecione aleatoriamente j_rand ∈ [0, 1]
        if j_rand < CR then
            u_{t,i} = x_{t,r1} + F(x_{t,r2} − x_{t,r3})
        else
            u_{t,i} = x_{t,i}
        end if
        avalia u_{t,i}
        if f(u_{t,1}) < f(x_{t,i}) then
            x_{t+1,i} = u_{t,i}
        else
            x_{t+1,i} = x_{t,i}
        end if
    end for
    t ← t+1
end while
sendo t a geração atual e X_t a população na geração t
formada pelos indivíduos x_{t,1}, x_{t,2}, ··· , x_{t,NP}.
```

3.1 Etapa I: Mutação

O processo de mutação visa introduzir novas informações no conjunto de dados através de alterações, muitas vezes aleatórias, nos elementos deste conjunto (Storn and Price, 1997a). Considere a solução apresentada na Equação 2:

$$V_{i,G} = \{v^1_{i,G}, \cdots, v^D_{i,G}\}, \qquad (2)$$

em que $v^{(\cdot)}_{i,G}$ representa o valor da *i-ésima* dimensão do indivíduo $V_{i,G}$. Segundo Qin et al. (2008), estes vetores podem sofrer mutações de diversas maneiras. Uma delas consiste em selecionar três indivíduos distintos, realizar a diferença entre dois destes indivíduos e então somar uma porcentagem dessa diferença a um terceiro indivíduo. Traduzindo esses passos matematicamente, tem-se a Equação 3:

$$V_{i,G} = X_{r^i_1,G} + F(X_{r^i_2,G} - X_{r^i_3,G}), \qquad (3)$$

em que F é uma constante cujo intuito é controlar a porcentagem de influência da mutação no indivíduo e $r^i_{(\cdot)}$ são índices que representam os indivíduos da população. Esses índices podem ser obtidos de diversas formas (Storn and Price, 1997a). Uma delas, abordada em Rocca et al. (2011), consiste em escolher três números inteiros e aleatórios compreendidos no intervalo [1, NP].

3.2 Etapa II: Cruzamento

O cruzamento tem o intuito de trocar informações entre indivíduos existentes (pais) e gerar novos indivíduos (filhos) (Storn and Price, 1997a). O processo de cruzamento utiliza o vetor original $X_{i,G}$, dado pela Equação 1, e o vetor mutante $V_{i,G}$, dado pela Equação 3. Nessa etapa são "misturados" os termos tanto de (1) quanto de (3), resultando no vetor apresentado na Equação 4:

$$U_{i,G} = \{u^1_{i,G}, \cdots, u^D_{i,G}\}, \qquad (4)$$

em que

$$u^j_{i,G} = \begin{cases} v^j_{i,G}, & \text{se}(\text{rand}_\text{j}[0,1] \leq \text{CR})\text{ou}(j = j_\text{rand}), \\ x^j_{i,G}, & \text{caso contrario},\ j = 1, 2, \cdots, D, \end{cases}$$

sendo CR uma constante que representa a probabilidade de cruzamento, geralmente entre 0,5 e 0,9 (Rocca et al., 2011), compreendida no intervalo [0,1]. Essa constante tem como finalidade controlar a quantidade de variáveis do vetor mutante.

3.3 Etapa III: Seleção

O processo de seleção é realizado entre os vetores $X_{i,G}$ (Equação 1) e $U_{i,G}$ (Equação 4). O valor de *fittnes* (nota que define a aptidão do indivíduo) de cada um dos vetores são comparados, sendo o melhor dentre eles (aquele que possui maior *fittnes*) selecionado para compor a próxima geração.

4. MÁQUINA DE VETORES DE SUPORTE

As máquinas de vetores de suporte (SVM) são fundamentadas na teoria de aprendizado estatístico. Essa teoria estabelece determinados princípios para que se possa obter classificadores com boa generalização (Lorena and de Carvalho, 2007). O trabalho de Haykin (1994) destaca que os resultados da aplicação de SVMs, em problemas de classificação, são muitas vezes superiores aos resultados obtidos por outros algoritmos baseados em técnicas diferentes, como as redes neurais, por exemplo.

4.1 Características das SVMs

De acordo com Smola et al. (2000) as principais características das SVMs, que justificam sua grande aplicabilidade, são as seguintes:

- **Capacidade de generalização:** Os classificadores gerados por uma SVM geralmente alcançam bons

resultados de generalização. A capacidade de generalização de um classificador é medida de acordo com sua eficiência em classificar dados diferentes daqueles utilizados na etapa de treinamento. Desse modo, evita-se o *overfitting*, que é uma situação onde o preditor se torna especialista nos dados de treinamento e apresenta baixos índices de desempenho quando aplicado a dados de validação e teste.

- **Robustez em grandes dimensões:** Redes SVMs são bastante robustas quando aplicadas a objetos de grandes dimensões, como imagens, por exemplo. Os classificadores obtidos por outras técnicas, como as redes neurais artificiais de múltiplas camadas (Haykin, 1994), geralmente resultam em *overfitting* quando aplicados a uma grande massa de dados.
- **Convexidade da função objetivo:** O problema de otimização das SVMs envolve uma função quadrática que possui um único mínimo global (Haykin, 1994). Essa condição garante a convergência da busca para o ponto ótimo da função objetivo. Isso confere às SVMs uma vantagem eminente em relação às redes neurais, por exemplo, que possuem mínimos locais na função objetivo.

4.2 Funções Kernel

As funções de kernel são utilizadas para mapear os vetores de características de entrada em um espaço de características de alta dimensão para problemas não linearmente separáveis. O aumento da dimensão do problema aumenta a probabilidade do mesmo se tornar linearmente separável, o que possibilita a utilização desta técnica (Gonçalves, 2015). A Figura 2 mostra um exemplo de como essas funções podem auxiliar na solução do problema mapeando-o para uma dimensão maior e transformando o problema em linearmente separável.

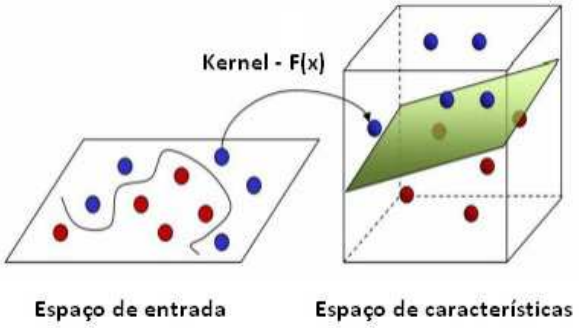


Figura 2. Transformação de um problema não linearmente separável em um problema linearmente separável. Adaptado de Gonçalves (2015).

5. METODOLOGIA

A metodologia proposta visa obter bons resultados de classificação utilizando uma quantidade reduzida de instâncias rotuladas, conhecendo, a priori, o espaço de domínio do conjunto de dados não rotulados. Desse modo, é possível reduzir o custo com a rotulação de dados, obtendo resultados satisfatórios com poucos dados rotulados.

Essa metodologia é baseada na modelagem do espaço de dados não rotulados em um problema de otimização. A modelagem utiliza a concentração de dados em uma dada região do espaço. Desse modo, utiliza-se o algoritmo de evolução diferencial para encontrar regiões com pouca concentração de dados e cujo a razão das distâncias entre os centros de massa do conjunto de dados seja próxima da unidade. Para exemplificar a metodologia, considere a Figura 3 que representa os dados de um problema de classificação. Este trabalho propõe distribuir soluções neste espaço, as quais, através de um algoritmo evolucionário, evoluem para a região de menor densidade de dados e que estejam entre os centros de massa das distribuições de dados. A Figura 4 mostra a população inicial do algoritmo evolucionário (DE).

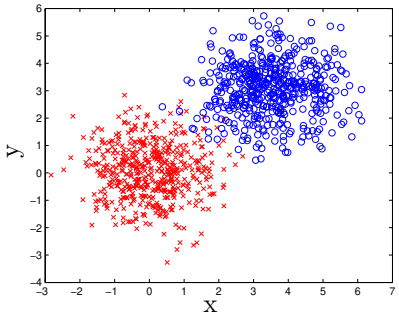


Figura 3. Distribuição de dados em um problema de classificação nas classes A ($\times$) e B ($\circ$).

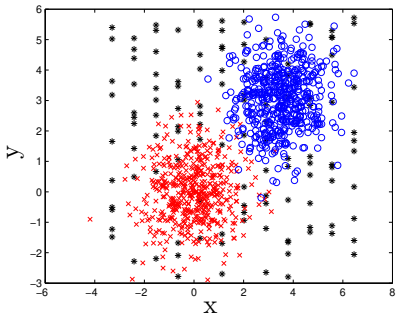


Figura 4. População inicial do algoritmo DE ($*$) juntamente com as classes A ($\times$) e B ($\circ$).

Para encontrar um conjunto capaz de separar as classes, algumas especificidades foram realizadas no algoritmo evolucionário DE. Inicialmente, utiliza-se as coordenadas dos centros das classes para definir qual eixo é o principal para a divisão das mesmas. Vale lembrar que para essa definição não é necessário conhecer os rótulos dos dados. O eixo escolhido é o que possui maior distância entre os centros das classes. A população inicial é gerada em partições de todo o domínio das instâncias. Nos eixos não principais, define-se um número de partições L, e então, gera-se uma malha nos eixos não principais, garantindo uma população esparsa.

No processo de evolução diferencial os deslocamentos foram feitos somente no eixo principal. Desse modo, a população evolui, porém continua sempre esparsa. Na Figura 4 a população evolui no eixo y, mantendo o eixo x fixo. Para a posição fixa, os dados são espaçados igualmente a partir da definição do número de partições desejadas.

O problema de otimização, necessário para definir o *fittnes* da solução do método DE, é formulado matematicamente como mostra a Equação 5, sendo x a variável de decisão do problema:

$$minimizar[P(x)] = \sum_{k=1}^{NS} F_k + R|D_i(x) - D_j(x)|, \quad (5)$$

$$F_k = \begin{cases} 1, \text{ se } y_k \in [x-L, x+L], \\ 0, \quad \text{caso contrario}, \end{cases}$$

em que NS é o número de dados do problema de classificação e $D_i(x)$ e $D_j(x)$ representam a distância entre a solução x do algoritmo DE e os centros de massa i e j dos dados do problema de classificação, respectivamente. O parâmetro R é uma constante de normalização da distância dos centros de massa, dado pela Equação 6:

$$R = |D_i - D_j|, \quad (6)$$

em que D_i e D_j representam a posição espacial dos centros de massa dos dados do problema de classificação. A variável F_k, por sua vez, é uma *flag* que indica se o dado y_k do problema de classificação está próximo da solução x. Para isso, utiliza-se o parâmetro L, que é calculado através da subtração dos dados mais distantes (no eixo fixo), dividido pelo número de partições desejadas.

Esse problema de otimização pode ser resolvido de diversas formas. Neste trabalho foram realizadas algumas alterações no processo de mutação do algoritmo DE como forma de evitar um grande "distanciamento" entre pai e filho gerados. A mutação foi realizada escolhendo aleatoriamente três indivíduos (r_j^i, $j = 1, 2, 3$) para o processo de deslocamento (Equação 3). Contudo, somente em uma variável de um indivíduo pai, sendo esta variável determinada de forma aleatória. O pseudo-código do processo de mutação proposto neste trabalho é mostrado no *Algorithm* 3.

Algorithm 3 Função Mutação Alternativa

Entrada:
PI - População pai
nvar - Número de variáveis do problema
tpop - Tamanho da população
Saída:
U - População após mutação
$F = 0.5$
for $i = 1$ *to tpop* **do**
  Pontos aleatórios e distintos: p_1, p_2, p_3
  $r_1 \leftarrow melhor(PI, p_1, p_2, p_3)$
  r_2 e $r_3 = piores(PI, p_1, p_2, p_3,)$
  $j_{rand} = aleatorio(nvar)$
  $f_i \leftarrow PI_i$
  $f_{i,j_{rand}} \leftarrow PI_{r_1,j_{rand}} + F(PI_{r_2,j_{rand}} - PI_{r_3,j_{rand}})$
end for
$U = PI \cup f$

A função *melhor* define, dentre os 3 indivíduos pais escolhidos, presentes na população PI, qual possui o melhor *fittnes*. A função *piores*, por sua vez, define os dois indivíduos com menor *fittnes*. Por fim, a função *aleatório* seleciona um número inteiro aleatório compreendido no intervalo [1, *nvar*].

Uma vez encontrada a solução utilizando o algorítimo DE, é possível definir um hiperplano separador que será o ponto de partida para o aprendizado ativo. A estratégia utilizada e definida por Settles (2008) consiste em consultar quais dados são incertos e, então, recorrer ao oráculo para definir o rótulo destes dados.

Neste trabalho, na primeira iteração, foram escolhidos dois dados que seriam rotulados pelo oráculo. Utiliza-se como critério de escolha os dados com maior proximidade ao hiperplano separador gerado pelo algoritmo DE. Estes dados são passados como parâmetro ao algoritmo SVM que realiza o treinamento. Como resultado, obtém-se um novo hiperplano separador.

Da segunda iteração em diante, um novo dado foi selecionado, utilizando o mesmo critério de escolha: a maior proximidade com o hiperplano separador atual. Juntamente com os dados já rotulados, esse novo dado foi passado como parâmetro ao algoritmo SVM que realiza um novo treinamento. Esse processo se repete até que seja atingido algum critério de parada.

Neste trabalho não foi definido um critério de parada, dado que a proposta consiste em verificar a eficiência do método. Tal eficiência será medida comparando o desempenho do algoritmo relativo à utilização de todos os dados para treinamento. Contudo, ao aplicar o método em problemas reais, pode-se recorrer à literatura para definir um critério de parada.

6. RESULTADOS

Os algoritmos de aprendizado ativo, geralmente, são avaliados por meio de uma curva de aprendizado (Settles, 2008). Nesta curva é mostrada a evolução do desempenho de aprendizado do algoritmo em função do aumento do número de instâncias rotuladas.

Neste trabalho, os resultados foram avaliados de duas maneiras. A primeira consiste em verificar, graficamente, o hiperplano separador obtido pelo algoritmo DE sem conhecimento de nenhum rótulo (Subseção 6.1). A segunda forma consiste na avaliação da curva de aprendizado em função do aumento do número de instâncias rotuladas, conforme apresentado em Settles (2008) (Subseção 6.2).

Para realização dos testes, o algorítimo DE foi executado utilizando os seguintes parâmetros: F=0,5, CR=0,55, população inicial aleatória de 100 indivíduos e número máximo de gerações igual a 130.

Com relação ao algoritmo de aprendizado ativo, não foi definido nenhum critério de parada. Desse modo, todos os dados foram utilizados para o treinamento da SVM. Tal consideração foi necessária para verificar a acurácia dos acertos nos dados de treinamento com a evolução do algoritmo de aprendizado ativo.

6.1 Resultados Gráficos

O primeiro teste realizado consiste em gerar 1000 dados em duas classes, a partir de uma distribuição normal. Tais classes estão espaçadas de forma a serem linearmente separáveis. A Figura 5 mostra esta distribuição de dados. Apesar dos dados serem representados por cores distintas, nenhuma informação sobre os rótulos das classes é conhecida pelo algoritmo DE. Os dados apresentados na Figura

5 são utilizados pelo algoritmo evolucionário DE com a finalidade de encontrar um conjunto solução para dar início ao algoritmo de aprendizado ativo. Após a finalização do algorítimo DE, realiza-se, então, uma regressão linear com a finalidade de determinar o primeiro hiperplano separador. Esse hiperplano é mostrado na Figura 6. Note que o hiperplano é capaz de separar as classes sem conhecê-las, dispensando qualquer esforço com a rotulação.

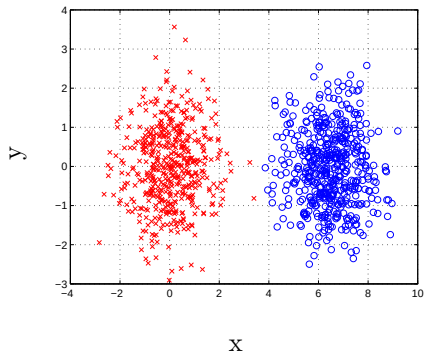


Figura 5. Distribuição de dados, de um problema de classificação, divididos em classes ($\times$ e $\circ$).

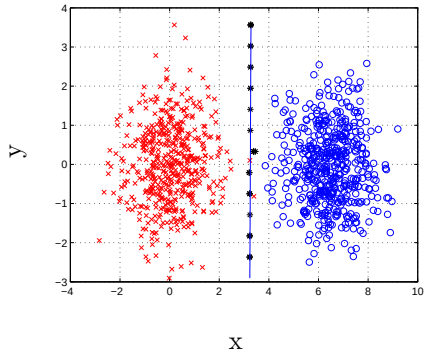
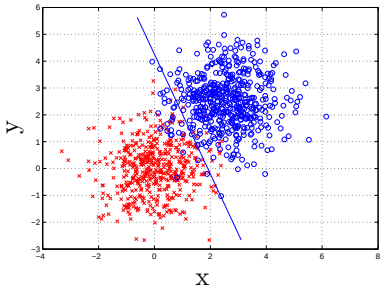


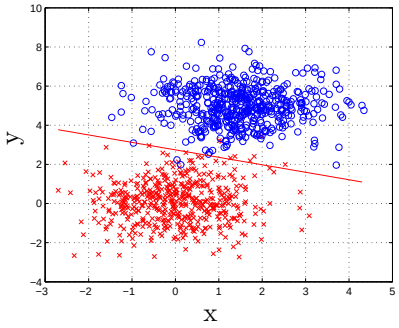
Figura 6. Solução encontrada pelo algoritmo DE. Classe A ($\times$), B ($\circ$) e conjunto solução estimado pelo algoritmo DE ($*$), sendo x o eixo princicpal. A linha contínua ($-$) representa o hiperplano de separação das classes, estimado por regressão linear, usando o conjunto solução.

O segundo experimento tem como finalidade verificar o comportamento do algoritmo DE com dados pertencentes a classes mais próximas, fazendo com que o problema não seja linearmente separável. Assim como no teste anterior, foram gerados 1000 dados em duas classes, a partir de uma distribuição normal. As Figuras 7a e 7b apresentam os resultados deste experimento. Observa-se que o hiperplano separador gerado pelo algoritmo DE também apresentou um resultado visualmente satisfatório, mostrando que o algoritmo é capaz de encontrar a região de separação das classes. Neste tipo de problema a rotulação de alguns dados pode ser útil no ajuste do hiperplano de separação das classes.

O terceiro e último experimento foi realizando utilizando dois conjuntos de dados com três dimensões e duas classes. Novamente foram utilizados 1000 dados gerados a partir de uma distribuição normal. As Figuras 8a e 8b apresentam



(a) Eixo principal: y.



(b) Eixo principal: x.

Figura 7. Conjunto de dados de um problema de classificação de duas classes não linearmente separáveis: A ($\times$), B ($\circ$) e o hiperplano separador encontrado pelo método DE ($-$).

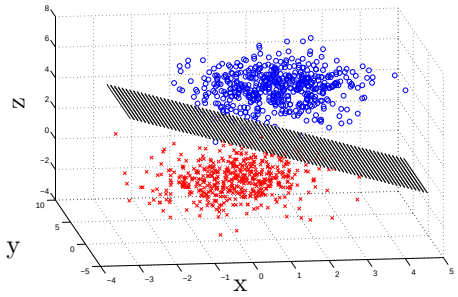
esses conjuntos de dados e os hiperplanos separadores estimados pelo algoritmo DE. Observa-se que os hiperplanos separadores encontram-se, em ambos os casos, próximo à região de separação das classes.

Os resultados dos testes gráficos mostram que o algoritmo DE é capaz de encontrar a região de separação para os problemas considerados, auxiliando, desse modo, na melhoria do desempenho do aprendizado ativo. Vale ressaltar, novamente, que as classes dos dados não são conhecidas pelo método DE.

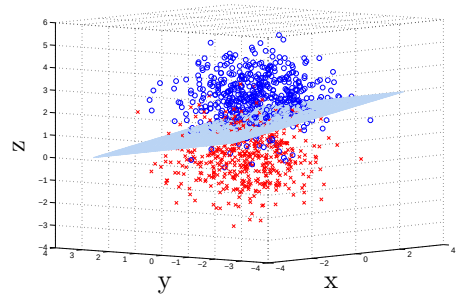
6.2 Desempenho da Curva de Aprendizado

Após a determinação dos hiperplanos que separam as classes, tem início o processo de aprendizado ativo, onde é definida uma estratégia de consulta. Deste modo, os dados escolhidos são rotulados e passados ao algoritmo de treinamento com a finalidade de gerar um novo classificador. A estratégia de consulta escolhida define que a instância mais incerta é aquela mais próxima ao hiperplano obtido pelo algoritmo evolucionário. Para determinar o hiperplano separador das classes utilizou-se um classificador baseado em SVM.

É importante destacar que neste trabalho os dados não foram separados em conjuntos de treinamento e teste. Todos os dados foram utilizados para treinamento, uma vez que o objetivo é verificar a evolução de desempenho do algoritmo com o incremento do número de instâncias rotuladas.



(a) Problema linearmente separável, sendo z o eixo principal.



(b) Problema não linearmente separável, sendo z o eixo principal.

Figura 8. Conjunto de dados em três dimensões de um problema de classificação de duas classes: A (×), B (○) e o hiperplano separador encontrado pelo método DE.

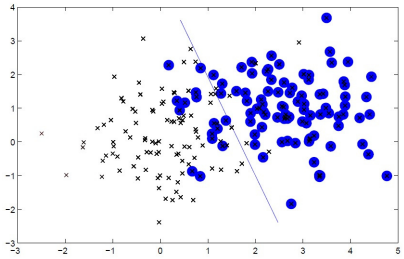
Os testes realizados a seguir tem a finalidade de analisar a curva de aprendizado do algoritmo de aprendizado ativo. Analisa-se o desempenho do método por meio da acurácia de acertos, de acordo com a relação apresentada na Equação 7

$$Acc = \frac{\#D_C}{\#D_T}, \tag{7}$$

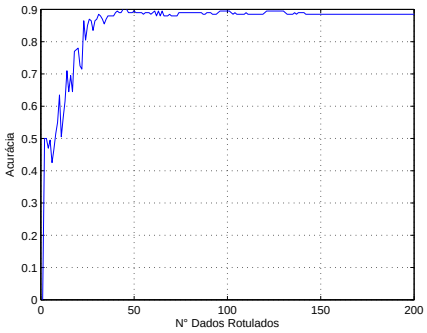
em que Acc é a acurácia de acertos, $\#D_C$ é o número de dados rotulados corretamente e $\#D_T$ é o número total de dados.

A Figura 9a mostra uma distribuição de classes, geradas a partir de uma distribuição normal com 200 dados e o hiperplano separador definido pelo algoritmo DE. A Figura 9b mostra a evolução da acurácia de acertos em função da quantidade de dados rotulados, após o treinamento, utilizando SVM. Percebe-se que, com menos de 20% dos dados rotulados, a acurácia de acertos se aproxima do resultado obtido com todos os dados rotulados.

As Figuras 10a e 10b apresentam, respectivamente, um conjunto de dados linearmente separável, gerados a partir de uma distribuição normal com 200 dados, e a evolução da acurácia de acertos. Nota-se que a obtenção de 100% de acertos na classificação foi obtida com menos de 10% dos dados rotulados. Este resultado mostra que o algoritmo DE representa uma boa alternativa para a escolha de dados a serem rotulados.

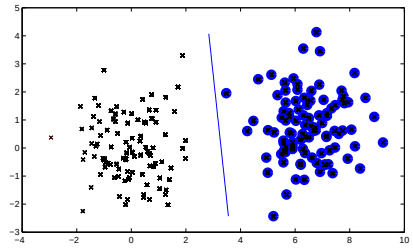


(a) Hiperplano separador obtido pelo algoritmo DE (-) em um conjunto de dados de duas classes não linearmente separáveis: A (×) e B (○).

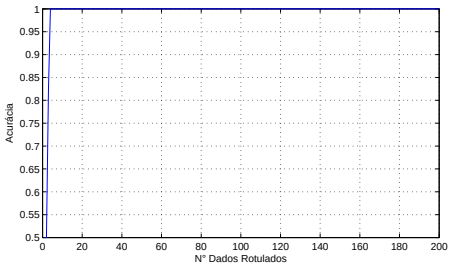


(b) Evolução da acurácia de acertos após o treinamento com os dados rotulados.

Figura 9. Evolução de desempenho para um problema não linearmente separável.



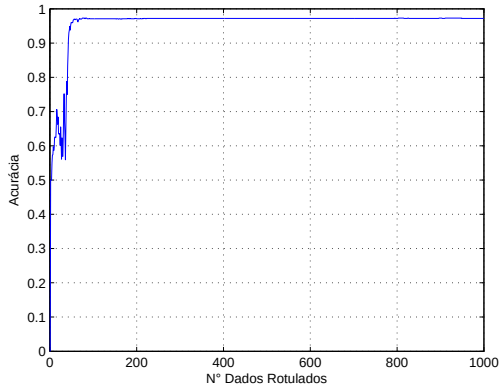
(a) Hiperplano separador obtido pelo algoritmo DE (-) em um conjunto de dados de duas classes linearmente separáveis: A (×) e B (○).



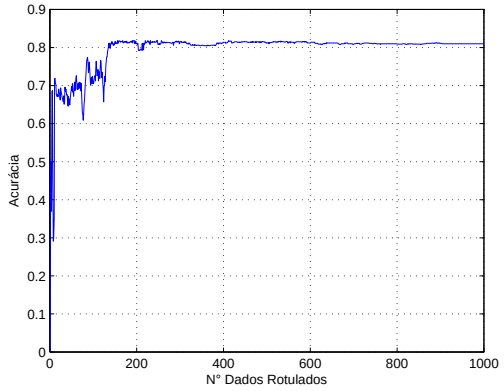
(b) Evolução da acurácia de acertos após treinamento com os dados rotulados.

Figura 10. Evolução de desempenho para um problema linearmente reparável.

Para os conjuntos de dados apresentados na Figura 7, também foram realizados os testes de evolução de desempenho da acurácia. Tais resultados são mostrados na Figura 11.



(a) Acurácia referente aos dados da Figura 7a.



(b) Acurácia referente aos dados da Figura 7b.

Figura 11. Evolução da acurácia de acertos após o treinamento com os dados rotulados.

As análises de evolução da acurácia de acertos mostram que, nos casos considerados, com menos de 20% dos dados rotulados, os resultados de acurácia se encontram próximos do valor máximo (com todos os dados rotulados).

7. CONCLUSÕES

Neste trabalho foi possível verificar o desempenho de uma técnica para determinação de dados iniciais para utilização de algoritmos de aprendizado ativo *pool − based*. Utilizando o algoritmo evolucionário de evolução diferencial, determinou-se uma população final próxima à região de separação de classes do problema de classificação de dados. Essa população tornou possível uma boa aproximação para o hiperplano separador das classes.

Os resultados mostram que, para os problemas abordados, a técnica é promissora, encontrando bons resultados de acurácia de acertos com poucos dados rotulados. Com a aplicação do aprendizado ativo, notou-se que o objetivo foi alcançado, uma vez que com uma pequena quantidade de dados rotulados foram obtidos resultados com uma alta acurácia. Além disso, os valores de acurácia variaram pouco com o aumento do número de dados rotulados, uma vez que a acurácia máxima foi obtida, no pior dos casos, com cerca de 15% dos dados rotulados.

Partindo deste ponto, sugere-se, em uma nova abordagem da técnica apresentada, elevar o números de dados utilizados na primeira iteração do algoritmo de aprendizado ativo, uma vez que, neste trabalho foram utilizados sempre dois dados na primeira iteração. Considerando que a metodologia apresentou bons resultado com a rotulação em torno de 15% dos dados, uma possibilidade de trabalho futuro seria utilizar uma quantidade menor de dados rotulados e avaliar o desempenho da técnica.

REFERÊNCIAS

Brinker, K. (2003). Incorporating Diversity in Active Learning With Support Vector Machines. *Proceedings of the Twentieth International Conference on Machine Learning (ICML): 2003*, 59–66.

Gonçalves, A.R. (2015). Máquina de Vetores de Suporte. *Notas de Aulas. Acessado em https://andreric.github.io/files/pdfs/svm.pdf.*

Haykin, S. (1994). *Neural Networks: a Comprehensive Foundation*. Prentice Hall PTR.

Holland, J.H. et al. (1992). *Adaptation in Natural and Artificial Systems: an Introductory Analysis With Applications to Biology, Control and Artificial Intelligence*. MIT press.

Lewis, D.D. and Ringuette, M. (1994). A Comparison of Two Learning Algorithms for Text Categorization. *Third Annual Symposium on Document Analysis and Information Retrieval*, 33, 81–93.

Lorena, A.C. and de Carvalho, A.C. (2007). Uma Introdução às Support Vector Machines. *Revista de Informática Teórica e Aplicada*, 14(2), 43–67.

Qin, A.K., Huang, V.L., and Suganthan, P.N. (2008). Differential Evolution Algorithm With Strategy Adaptation For Global Numerical Optimization. *IEEE Transactions on Evolutionary Computation*, 13(2), 398–417.

Rocca, P., Oliveri, G., and Massa, A. (2011). Differential Evolution as Applied to Electromagnetics. *IEEE Antennas and Propagation Magazine*, 53(1), 38–49.

Settles, B. (2008). *Curious Machines: Active Learning With Structured Instances*. Ph.D. thesis, University of Wisconsin-Madison.

Smola, A.J., Bartlett, P.J., Schuurmans, D., Schölkopf, B., Jordan, M.I., et al. (2000). *Advances in Large Margin Classifiers*. MIT press.

Storn, R. and Price, K. (1997a). Differential Evolution - A Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces. *Journal of Global Optimization*, 11, 341–359.

Storn, R. and Price, K. (1997b). Differential Evolution: a Simple and Efficient Heuristic For Global Optimization Over Continuous Spaces. *Journal of Global Optimization*, 11(4), 341–359.

Vapnik, V.N. (1999). An Overview of Statistical Learning Theory. *IEEE Transactions on Neural Networks*, 10(5), 988–999.

Zhu, X.J. (2005). Semi-Supervised Learning Literature Survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.