Estratégias de Maximização de Margem para o Perceptron

Matheus Bitarães de Novaes

I. Introdução

Este artigo tem como objetivo apresentar o modelo do perceptron simples bem como algumas estratégias para a maximização da margem obtida. As estratégias de maximização de margem tem como o objetivo encontrar a superfície de separação que divida o plano da forma mais generalista possível em dados linearmente separáveis.

o trabalho contará com uma revisão da literatura, implementação de três algoritmos baseados no perceptron, sendo um deles uma proposição de um modelo de comitê para resolução do problema de maximização de margem. será feita uma comparação estatística do desempenho destes algoritmos de acordo com 5 *datasets*.

II. Revisão de Literatura

O Perceptron simples [1] é um classificador linear caracterizado por uma matriz de pesos w, que são multiplicados às entradas X e submetidos a uma função de ativação definida por uma função degrau [2]. Após o treinamento, os pesos w definem a equação de um plano que irá separar os dados das classes do problema.

O processo de ajuste dos pesos w do Perceptron dá-se pela correção dos erros da saída do modelo em comparação com os dados utilizados para o treinamento. Durante o processo os pesos serão corrigidos quando a saída do modelo for diferente da saída esperada, conforme a equação abaixo:

$$w(t+1) = w(t) + \eta e(t) x(t)$$

onde $w(t)$, $e(t)$ e $x(t)$ representam, respectivamente, os valores do vetor de pesos, do erro e do vetor de entrada no instante t [2].

Em problemas linearmente separáveis, podem existir infinitas equações que definam um plano que separe os pontos de diferentes classes. É necessário escolher um plano que separe corretamente os pontos e que esteja posicionado de forma a ser o mais equidistante das duas classe quanto possível. Estratégias de maximização de margem são, portanto, utilizadas para tentar-se encontrar este plano, ou uma aproximação dele, de maneira eficiente. Um artigo que propõe uma abordagem para isso é o de Yoav Freund e Robert E. Schapire [3].

O trabalho propõe o uso de um algoritmo chamado de *voted-perceptron*, que combina o Perceptron com uma variação do algoritmo *leave-one-out* de Helmbold e Warmuth [4]. O algoritmo proposto armazena os vetores gerados durante o treinamento e cria pesos para privilegiar vetores que possuem maior taxa de acerto durante o processo. É realizado um comitê com todos os vetores e cada um vota em uma resposta. Os resultados são ponderados pelo peso de cada vetor. O pseudocódigo pode ser visto na figura 1.

Training
Input: a labeled training set $\langle(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\rangle$
number of epochs T
Output: a list of weighted perceptrons
$\langle(\mathbf{v}_1, c_1), \ldots, (\mathbf{v}_k, c_k)\rangle$

- Initialize: $k := 0$, $\mathbf{v}_1 := \mathbf{0}$, $c_1 := 0$.
- Repeat T times:
  - For $i = 1, \ldots, m$:
    - Compute prediction: $\hat{y} := \text{sign}(\mathbf{v}_k \cdot \mathbf{x}_i)$
    - If $\hat{y} = y$ then $c_k := c_k + 1$.
      else $\mathbf{v}_{k+1} = \mathbf{v}_k + y_i \mathbf{x}_i$;
      $c_{k+1} = 1$;
      $k := k + 1$.

Prediction
Given: the list of weighted perceptrons:
$\langle(\mathbf{v}_1, c_1), \ldots, (\mathbf{v}_k, c_k)\rangle$
an unlabeled instance: $\mathbf{x}$
compute a predicted label $\hat{y}$ as follows:

$$s = \sum_{i=1}^{k} c_i \ \text{sign}(\mathbf{v}_i \cdot \mathbf{x}); \qquad \hat{y} = \text{sign}(s) \ .$$

Fig. 1. Algoritmo *Voted-perceptron*[4]

Outra abordagem para o problema pode ser encontrada no trabalho de Saul Leite e Raul Fonseca[5]. Os autores propõem um conjunto de dois algoritmos chamados *Fixed Margin Perceptron* (FMP) e *Incremental Margin algorithm* (IMA), que busca encontrar a solução para um problema linearmente separável dada uma margem fixa. O algoritmo foi desenvolvido com o intuito de evitar resolução do problema de programação quadrática que é utilizado ao se obter a margem máxima calculada pela SVM. Os pseudocódigos destes algoritmos podem ser vistos nas figuras 2 e 3

Algorithm 1. Primal FMP algorithm.

1: **Input**: z_m, $\boldsymbol{w}_{\text{init}}$, γ_f, η, T_MAX
2: $\boldsymbol{w}^0 \leftarrow \boldsymbol{w}_{\text{init}}$, $t \leftarrow 0$
3: **repeat**
4: **for** $(i = 1, \ldots, m)$ **do**
5: **if** $(y_i \langle \boldsymbol{x}_i, \boldsymbol{w}^t \rangle < \gamma_f \|\boldsymbol{w}^t\|)$ **then**
6: $\boldsymbol{w}^{t+1} \leftarrow \boldsymbol{w}^t \lambda_t + \eta y_i \boldsymbol{x}_i$
7: $t \leftarrow t + 1$
8: **end if**
9: **end for**
10: **until** (no mistakes were made) or $(t > T\_MAX)$
11: return $\boldsymbol{w}^t$

Fig. 2. Algoritmo *Fixed Margin Perceptron* (FMP) [5]

Algorithm 2. Incremental margin algorithm.

1: **Input**: z_m, η, δ, T_MAX
2: $\boldsymbol{w} \leftarrow 0$, $\gamma_{\mathrm{f}} \leftarrow 0$
3: **repeat**
4: $\quad \boldsymbol{w} \leftarrow \mathrm{FMP}(z_m, \boldsymbol{w}, \gamma_{\mathrm{f}}, \eta, T\_MAX)$
5: $\quad \gamma_{\mathrm{f}} \leftarrow \max((\gamma^{+}(\boldsymbol{w}) + \gamma^{-}(\boldsymbol{w}))/2, (1+\delta)\gamma_{\mathrm{f}})$
6: **until** the convergence of FMP in T_MAX iterations is not achieved
7: return last feasible $\boldsymbol{w}$

Fig. 3. Algoritmo *Incremental Margin algorithm* (IMA)

Para este trabalho, serão implementadas duas estratégias de maximização de margem. A primeira é o algoritmo *voted-perceptron*, descrito anteriormente e definido em [4]. A segunda estratégia é um comitê de perceptrons, onde o resultado majoritário entre 5 perceptrons definirá a saída do modelo. Espera-se que, com esta estratégia, a saída do modelo seja mais genérica e assertiva do que a avaliação de apenas 1 perceptron.

III. Metodologia

Para este trabalho serão utilizados os seguintes grupos de dados:

- *Two Gaussians Dataset*: *Dataset* fictício, artificialmente gerado, com centros em (2,2) e (4,4)
- *Wine Dataset* [6]: *Dataset* com características químicas de vinhos fabricados na mesma região da Itália, porém vindos de três cultivadores diferentes. É uma base de dados com 13 atributos e 3 classes. Para este trabalho, uma das classes foi removida. Desta forma, o problema tornou-se uma classificação binária.
- *Pima Indians Dataset* [7]: Este é um *dataset* com 8 atributos de 768 casos clínicos e duas classes.
- *Iris Dataset* [8]: Este *dataset* contém três classes de 50 instâncias cada, onde cada classe é um tipo de planta iris. Este grupo de dados é composto de 4 atributos que descrevem as dimensões de cada planta. Como este *dataset* possui 3 classes, foi necessário a remoção de uma das classes para que se tornasse um problema de classificação binária.
- *Cervical Cancer Dataset* [9]: Este *dataset* foi coletado no *Hospital Universitario de Caracas* em Caracas, Venezuela e contém informações demográficas, hábitos e histórico medico de 858 pacientes. O *dataset* possui alguns dados vazios que precisam ser tratados.

Todos os grupos de dados serão normalizados e os que possuem dados faltantes serão tratados. As linhas que possuem algum campo faltante serão removidas do *dataset*.

Após o pré-tratamento e normalização, os três algoritmos serão treinados e avaliados com uma divisão de 70% dos dados para treinamento e 30% dos dados para teste. Este processo será repetido por 50 vezes e os resultados serão comparados.

As comparações entre os *datasets* serão feitas utilizando os testes de Friedman para identificação de diferença estatisticamente significativa em ao menos um dos modelos e, após isto, será realizado o teste de Nemenyi para identificação dos modelos que diferem entre si. O teste de Friedman [10] foi escolhido pois é um teste que não depende que as amostras possuam distribuição normal, que é o nosso caso. Ao se realizar o teste de *shapiro-wilk* para a distribuição de acurácias do primeiro dataset (*iris dataset*), foi identificado que não seguiam a distribuição normal e, portanto, é necessário seguir com um teste não paramétrico, que é o caso do teste de Friedman.

Serão avaliadas as diferenças em valores de acurácia e tempo de treinamento.

IV. Resultados

As três abordagens utilizam o *Perceptron* com as seguintes configurações:

- Número Máximo de Epocas: 100
- η: 0.01
- tolerância: 0.01

A. Duas Gaussianas

Para o *dataset* das duas gaussianas, podemos ver pelo teste estatístico que não há diferença significativa entre as acurácias, porém, nota-se um menor tempo de treinamento para o algoritmo **perceptron**, como pode-se notar pelas figuras 4 e 5

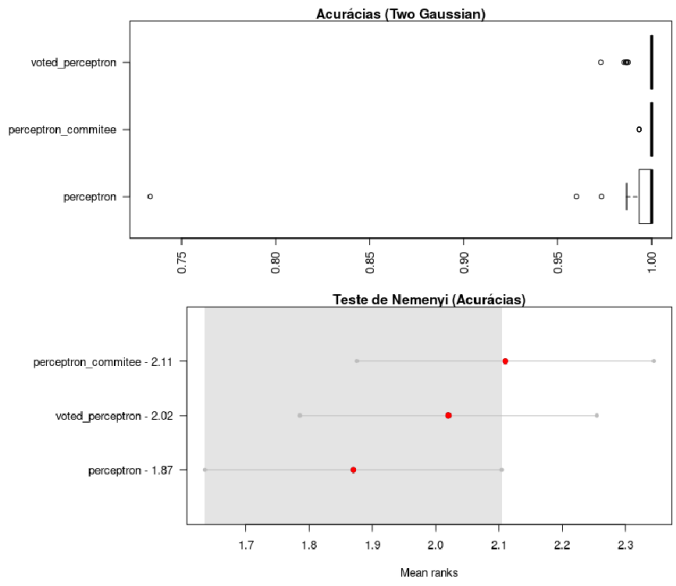

Fig. 4. Boxplot das acurácias dos 3 algoritmos para o dataset de duas gaussianas

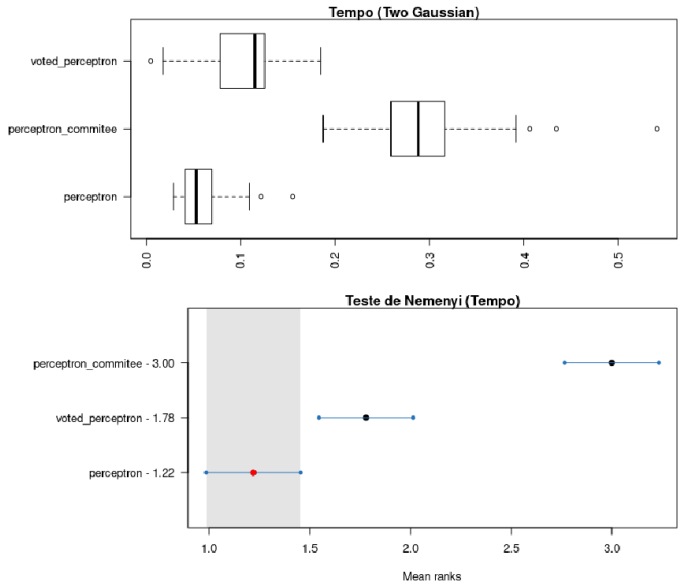

Fig. 5. Boxplot dos tempos de execução dos 3 algoritmos para o dataset de duas gaussianas

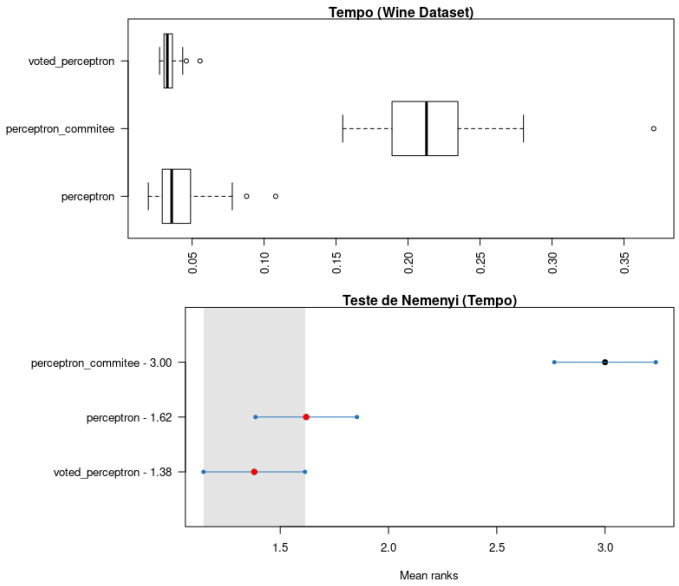

Fig. 7. Boxplot dos tempos de execução dos 3 algoritmos para o *Wine Dataset*

B. *Wine Dataset*

Para o *Wine Dataset*, nota-se que o teste estatístico indica uma melhor performance para o algoritmo **perceptron_commitee**, que também possui o maior tempo de treinamento (figuras 6 e 7)

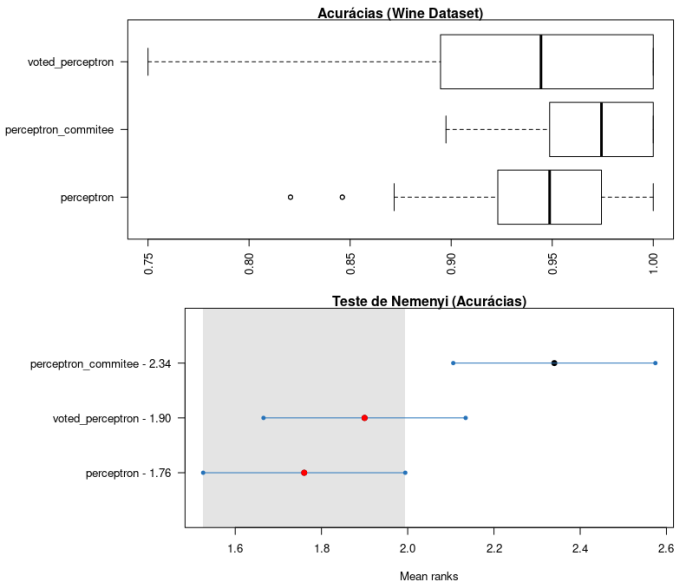

Fig. 6. Boxplot das acurácias dos 3 algoritmos para o *Wine Dataset*

C. *Pima Indians Dataset*

Pelas figuras 8 e 9 pode-se observar que, para o *Pima Indians Dataset* o perceptron e o perceptron_commitee tiveram as melhores performances. Olhando também para as comparações de tempo de treinamento, o **perceptron** é portanto o algoritmo que apresenta melhor performance considerando os indicadores de acurácia e tempo de treinamento.

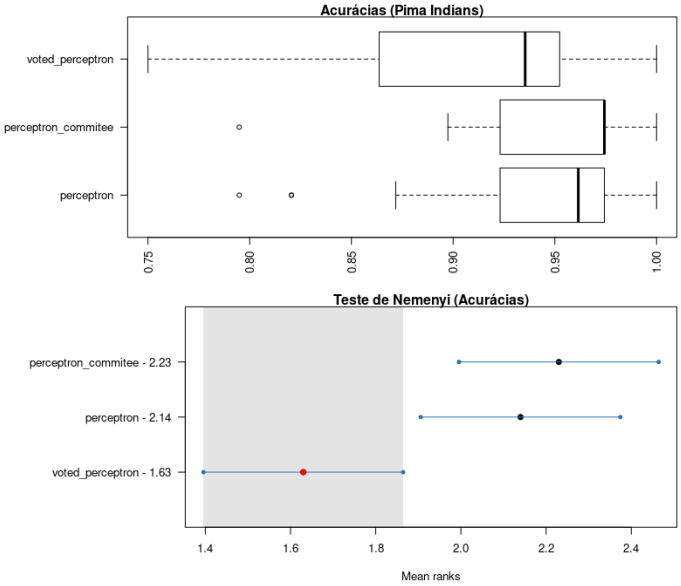

Fig. 8. Boxplot das acurácias dos 3 algoritmos para o *Pima Indians Dataset*

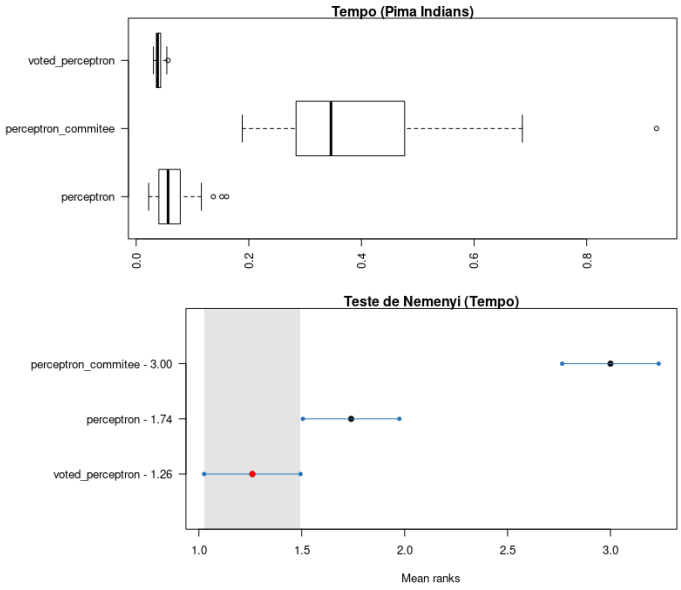

Fig. 9. Boxplot dos tempos de execução dos 3 algoritmos para o *Pima Indians Dataset*

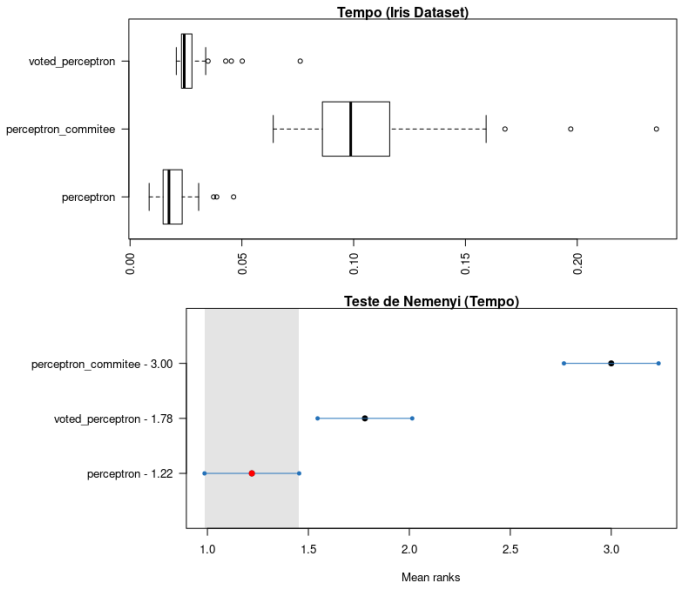

Fig. 11. Boxplot dos tempos de execução dos 3 algoritmos para o *Iris Dataset*

D. Iris Dataset

Para o *Iris Dataset* pode-se observar pelas figuras 10 e 11 que não há diferença estatisticamente significativa entre as acurácias. Porém, nota-se que o tempo de treinamento do **perceptron** é inferior aos demais, sendo este algoritmo portanto o indicado para este problema.

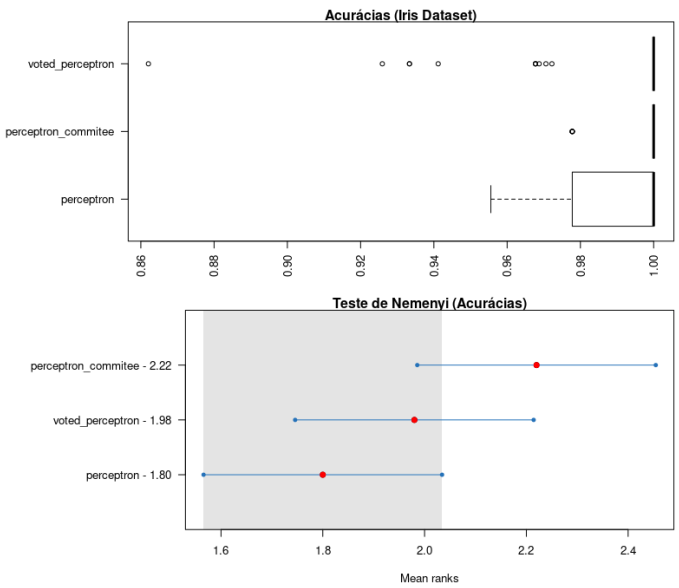

Fig. 10. Boxplot das acurácias dos 3 algoritmos para o *Iris Dataset*

E. Cervical Cancer Dataset

Para o *Cervical Cancer Dataset* não é possível encontrar diferença estatisticamente significativa entre as acurácias encontradas, porém o **voted_perceptron** apresentou menor tempo de treinamento. Portanto, este é o algoritmo indicado para a resolução do problema (figuras 12 e 13).

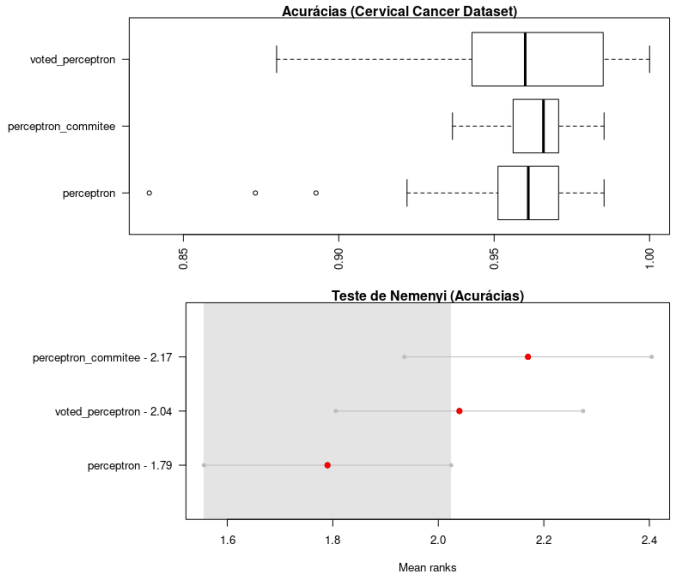

Fig. 12. Boxplot das acurácias dos 3 algoritmos para o *Cervical Cancer Dataset*

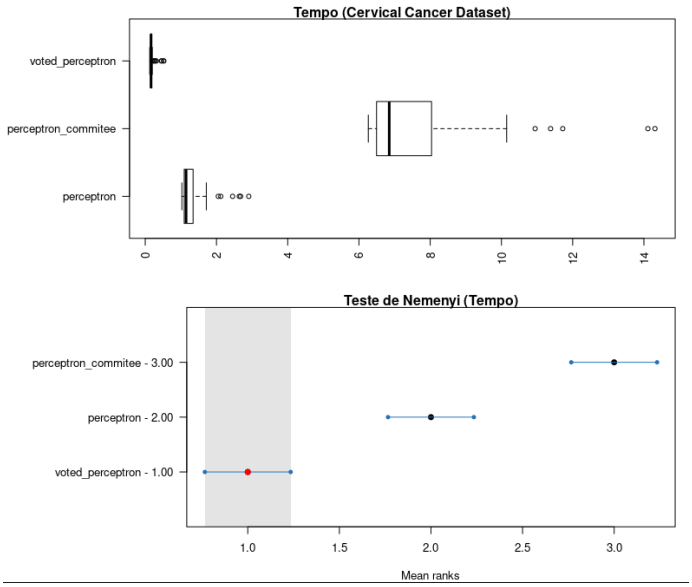

Fig. 13. Boxplot dos tempos de execução dos 3 algoritmos para o *Cervical Cancer Dataset*

V. Discussões

Através dos resultados e testes estatísticos realizados na sessão anterior, pode-se observar que o **perceptron** apresentou melhor resultado em 3 algoritmos, o **voted_perceptron** apresentou melhor resultado em 1 algoritmo e o **perceptron_commitee** apresentou melhor resultado em 1 algoritmo também. Os dois modelos de maximização de margem, conforme o esperado, demoram mais para serem treinados e, em 3 dos 5 casos, não apresentaram melhora significativa em comparação com o perceptron simples. Justifica-se o uso dos modelos de maximização de margem em um contexto onde o erro seja caro e o tempo de treinamento não seja um valor importante a ser considerado.

VI. Conclusões

Através deste trabalho pode-se notar a influencia que algoritmos de maximização de margem podem exercer no aumento da acurácia de um modelo, em comparação com o perceptron simples. Foi possível notar também o efeito dessas estratégias no tempo de treinamento do modelo. Os algoritmos de maximização de margem apresentaram acurácias semelhantes e as vezes superiores ao perceptron simples, porém acarretam um maior tempo de treinamento. Portanto, são indicados em contextos onde o aumento da acurácia é mais importante que o aumento no tempo de treinamento

References

[1] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[2] A. P. Braga, *Aprendendo com Exemplos: Princípios de Redes Neurais Artificiais e de Reconhecimento de Padrões*. Escola de Engenharia UFMG, 2021, vol. 01.

[3] Y. Freund and R. E. Schapire, "Large margin classification using the perceptron algorithm," *Machine learning*, vol. 37, no. 3, pp. 277–296, 1999.

[4] D. P. Helmbold and M. K. Warmuth, "On weak learning," *Journal of Computer and System Sciences*, vol. 50, no. 3, pp. 551–573, 1995.

[5] S. C. Leite and R. F. Neto, "Incremental margin algorithm for large margin classifiers," *Neurocomputing*, vol. 71, no. 7-9, pp. 1550–1560, 2008.

[6] P. Forina M. et al, *UCI machine learning repository - wine dataset*, 1991. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/wine.

[7] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *AAAI*, 2015. [Online]. Available: https://networkrepository.com.

[8] R. Fisher, *UCI machine learning repository - iris dataset*, 1936. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/iris.

[9] A. W. Sobar Rizanda Machmud, *UCI machine learning repository - cervical cancer behavior risk dataset*, 2019. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Cervical+Cancer+Behavior+Risk.

[10] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the american statistical association*, vol. 32, no. 200, pp. 675–701, 1937.