Redes Neurais Artificiais: Artigo 02

Leonam R. S. Miranda, Departamento de Engenharia Elétrica, Universidade Federal de Minas Gerais

***Resumo*—Neste trabalho foi feito um estudo comparativo entre diferentes modelos de redes neurais, em diferentes conjuntos de dados.**

***Palavras-chave*—Redes Neurais, Perceptron, ELM, Adaline, RBF, Hebb, Valição Cruzada, Regularização**

I. Introdução

Neste trabalho será feito um estudo comparativo entre diferentes modelos de redes neurais, Perceptron, Adaline, ELM, RBF e ELM Hebiano (Euler) [1]. A utilização de cada modelo dependerá da base em questão, ou seja, em problemas de classificação, por exemplo, não deve ser utilizado o Adaline. Assim serão avaliados 6 problemas de classificaçõ e 2 problemas de regressão. A escolha dos conjuntos de dados foi feita visando obter alguma diversidade quanto ao número de variáveis, número de amostras, dados desbalanceados, etc. A maior parte das bases de dados utilizados neste trabalho se encontram no reposiório de aprendizagem de máquina da UCI [2]. Para os problemas de classificação a acurácia e a área sobre a curva ROC (receiver operating characteristic) AUC (area under curve) [3], devido ao desbalanceamento de algumas das bases utilizadas, foram adotadas como metricas de performance. Para os problemas de regressão foi utilizado o coeficiente de determinação, denotado como R^2 como métrica de performance [4]. Para cada modelo, em cada uma das bases de dados, será utilizado validação cruzada para selecionar os hiperparâmetros, visando encontrar o modelos onde o erro esperado seja baixo, sem sobreajuste [5]. Na medida do possível, serão utilizadas técnicas de regularização.

II. Modelos de Redes Neurais Utilizados

A. Aprendizagem Hebbiana

O aprendizado Hebbiano é uma forma de aprendizado não supervisionado cujo resultado é proporcional ao produto cruzado dos padrões pelos seus rótulos. Essa forma de aprendizado não se baseia em correção do erro, não exigindo, portanto, a comparação do rótulo desejado pelo obtido. Um termo residual devido a não ortogonalidade dos dados estará presente na maioria das aplicações

No aprendizado de redes neurais artificiais, a atualização dos pesos w_{ij} que conectam neurônios i e j de acordo com a regra de Hebb [6] é proporcional ao produto cruzado de seus valores de ativação para cada associação entrada e saída k ou, em outras palavras, $w_{ij} \propto x_{ik} y_{jk}$. A regra pode ser escrita na seguinte forma matricial:

$$w = X^T y \tag{1}$$

Conforme [1], como $\hat{y} = XW$. Ao substituir na equação 1 tem-se que $\hat{y} = XX^T y$. Para recuperar perfeitamente y deve-se ter $XX^T = I$. Se essa condição é obtida o erro de treinamento é zero, já que $\hat{y} = y$, e a regra de aprendizado seria equivalente ao método da pseudoinversa. Nessa situação X é uma base ortonormal e $X^T = X^{-1}$ [7]. . Entretanto, na maioria das situações reais X não será uma base ortonormal e um termo residual devido ao produto XX^T causará um deslocamento de $\hat{y}$ em relação a y.

Como $\hat{y}_k = \sum_{j=1}^{N} x_k^T x_j y_j$, o termo correspondente a $j = k$ pode ser separado da soma o que leva a equação 2 . Na situação particular em que os dados de entrada são normalizados, ou seja $x_k^T xk = 1$, a Equação 2 pode ser simplificada na Equação 3 [8]:

$$\hat{y}_k = x_k^T x_k y_k + \sum_{j=1, j\neq k}^{N} x_k^T x_j y_j \tag{2}$$

$$\hat{y}_k = y_k + \sum_{j=1, j\neq k}^{N} x_k^T x_j y_j \tag{3}$$

O somatório $\sum_{j=1, j\neq k}^{N} x_k^T x_j y_j$ é chamado de ***crosstalk***. O ***crosstalk*** é ineente ao aprendizado Hebbiano e é devido à não ortogonalidade dos dados de entrada; Ao realizar uma seleção dos padrões mais relevantes a serem aprendidos gera um efeito de regularização, diminuindo o sobreajuste. Na tese do EULer [1] é proposto estratégias de aprendizado ativo para seleção de padrões para serem aplicados no aprendizado Hebbiano.

B. Perceptron

Em 1959, a descoberta de células simples e células complexas que constituem o córtex visual primário pelos ganhadores do Prêmio Nobel Hubel e Wiesel [**?**] teve uma ampla influência em muitos campos, incluindo o design de redes neurais. Frank Rosenblatt estendeu o neurônio McCulloch Pitts ([9] [10]) usando o termo Mark I Perceptron, que recebia entradas, gerava saídas e tinha lógica de limiar linear [11].

Um perceptron é uma rede neural Feed Forward assim como o neurônio de Mcculloch and Pitts (onde a informação é sempre transmitida na direção da camada de entrada para a camada de saída) e tinha lógica de *thresholding* linear. Desta forma, o modelo de Mcculloch e Pitts pode ser considerado como o tipo mais simples de perceptron.

Concretamente, Rosenblatt mostrou que um perceptron de uma camada é capaz de aprender muitas funções práticas. Ele propôs uma regra de aprendizado para

o perceptron, chamada regra do perceptron. Consideremos o caso mais simples de um perceptron de uma camada composto por um único neurônio, ou seja, o modelo proposto por McCulloch e Pitts. Se certos pares de entrada e saída correspondente forem conhecidos, $D_N = (x_1, d_1), (x_2, d_2), ..., (x_N, d_N)$, então, em um determinado padrão de entrada x_k do conjunto de dados de entrada, a regra perceptron atualiza os pesos da rede $\mathbf{w} = [w_0, w_1, ..., w_M]^T$ da seguinte forma:

$$w(k+1) = w(k) + \eta(d_k - o_k)x_k \tag{4}$$

O parâmetro η controla os valores de magnitude de atualização e, portanto, a velocidade de convergência do algoritmo. É chamado de *taxa de aprendizagem* e geralmente assume valores na faixa entre 0 e 1. O conjunto D_N é chamado de conjunto de treinamento, d_k é a saída desejada e o_k é a saída obtida.

Se a separabilidade linear é realizada pelo conjunto de dados de treinamento, Rosenblatt mostrou que o algoritmo sempre converge em um número finito de passos, independente do valor. Pelo contrário, se o problema não for linearmente separável, terá que ser forçado a parar, pois sempre haverá pelo menos um padrão classificado erroneamente.

É interessante notar que se a taxa de aprendizado tiver um valor próximo a 0, os pesos terão uma pequena variação a cada nova entrada, e o aprendizado é lento; se o valor for próximo a 1, pode haver grandes diferenças entre os valores de peso para uma iteração e a seguinte, reduzindo a influência das iterações anteriores e o algoritmo não pode convergir. Este problema é denominado *instabilidade*.

Também no início da década de 1960, Widrow e Hoff [12] realizaram várias demonstrações em sistemas do tipo perceptron, chamados de *ADALINE* ("Elementos LINear ADAptativos"), propondo uma regra de aprendizagem chamada de *algoritmo LMS* (algoritmo "Least Mean Square") ou algoritmo Widrow-Hoff. Essa regra minimiza a soma dos erros quadrados entre a saída desejada e a saída fornecida pelo perceptron. Ou seja, ele minimiza a função de erro:

$$E(w) = \frac{1}{2}\sum_{j=1}^{N}(d_j - z_j)^2 \tag{5}$$

Quando o gradiente para w é aplicado na Equação (4) e atualizado na direção oposta ao gradiente, a regra LMS é obtida.

$$w(k+1) = w(k) + \eta\sum_{j=1}^{N}(d_j - z_j(k))x_j \tag{6}$$

onde $x_j(k) = \mathbf{w}^T(k)x_j$. Essa versão de LMS é usualmente substituida por uma "aproximação estocástica"conforme a equação 7.

$$w(k+1) = w(k) + \eta(d_k - z_k)x_k \tag{7}$$

Ao contrário da regra do perceptron, a aplicação do LMS oferece resultados razoáveis (o melhor que pode ser alcançado por meio de um discriminador linear) quando o conjunto de treinamento não é linearmente separável.

Neste trabalho foi utilizado os algoritmos do Perceptron e Adaline disponíveis na biblioteca em python do scikit learn [13]. Em ambos os algorítmos foi utilizado a regularização L2 (regularização de Ridge), penalizando a soma ao quadrado dos pesos, alterando a função de custo para a equação 8, obtendo assim modelos menos propensos a sobre-ajuste. Assim para cada modelo Perceptron e Adaline utilizado neste trabalho, deve-se definir um fator de regularização (λ).

$$E(w) = \frac{1}{2}\sum_{j=1}^{N}(d_j - z_j)^2 + \lambda\frac{1}{2}\sum w^2 \tag{8}$$

C. Máquinas de Aprendizado Extremo (ELMs)

Uma das principais dificuldades em se treinar redes neurais artificiais de duas camadas do tipo Multilayer Perceptron (MLP) é o cálculo dos pesos da camada escondida. Em geral, abordagens tradicionais realizam a retropropagação do erro de treinamento através das camadas da rede neural artificial. Entretanto, é sabido desde a década de 1990 que os parâmetros da camada escondida (pesos e bias) podem ser definidos aleatoriamente, bastando realizar o treinamento dos parâmetros (pesos) da camada de saída [14]. Essa forma de treinamento se tornou popular somente após o ano de 2006 com o trabalho de Huang et al. [15] que apresentou as Máquinas de Aprendizado Extremo (Extreme Learning Machine - ELM). A Figura 1 apresenta a topologia típica das ELMs.

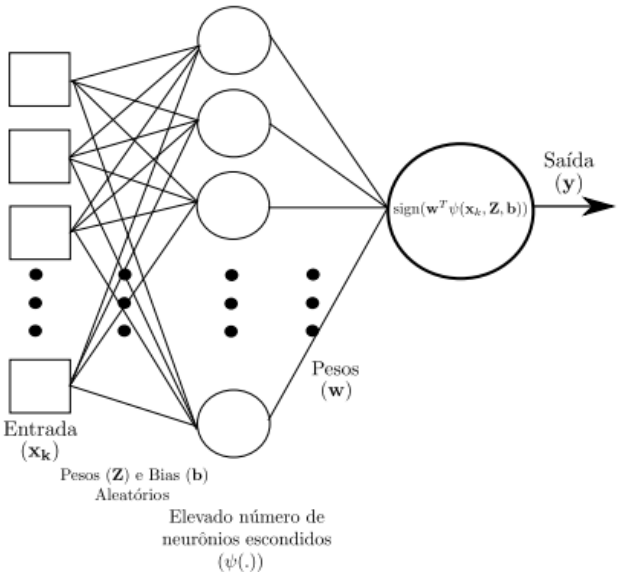

Figura 1. Topologia típica das ELMS

A função $H = \psi(x, Z, b)$, com argumentos x, matriz de parâmetros Z e vetor de bias b, mapeia cada uma das linhas de X nas linhas da matriz de mapeamento H de dimensão $N\ x\ p$ (Sendo N o número de amostras e p o número de neurônios escondidos).

Neste trabalho os modelos **ELM** e **ELM Hebbiano**, os elementos Z e b foram escondidos aleatóriamente. Entretanto **ELM Hebbiano** foi aplicado a regra de Hebb na camada de saída. O modelo **RBF** é um ELM em que são utilizadas funções radiais, ou funções de agrupamento como o *k-médias* [16] para determinar a matriz H.

O vetor w de dimensão $p \times 1$ contém os parâmetros do separador linear na camada escondida e é obtido pela solução de um sistema de equações lineares de N equações:

$$Hw = y \tag{9}$$

A solução de norma mínima desse sistema de equações é dado por:

$$w = H^{+}y \tag{10}$$

em que H^{+} é a *pseudoinversa de Moore-Penrose*.

Neste trabalho foi aplicado o método de **regularização L2** (*Ridge Regression*), de forma a obter modelos menos complexos, evitando que ocorra sobre-ajuste, coforme proposto em [17]. Assim em todos os modelos ELM deve-se definir um fator de regularização (λ), além do número de neurônios na camada intermediária (parâmetro p).

III. Seleção de modelos

Para cada um dos modelos há hiperparâmetros que devem ser escolhidos antes de treiná-los. Para cada um dos conjuntos de dados ao alterar os hipeparâmetros dos modelos são obtidas valores diferentes de acurácia. Assim, a escolha dos conjunto de hiperparâmetros para cada modelo em cada um dos conjuntos de dados foi feita através de busca intensiva em grade [18].

Aprender os parâmetros de uma função de previsão e testá-la com os mesmos dados é um erro metodológico: um modelo que apenas repetisse os rótulos das amostras que acabou de ver teria uma pontuação perfeita, mas não conseguiria prever nada de útil para dados ainda não vistos. Essa situação é chamada de overfitting. Para evitá-lo, é prática comum, ao realizar um experimento de aprendizado de máquina (supervisionado), manter parte dos dados disponíveis como um conjunto de testes X_test, y_test. Observe que a palavra “experimento” não tem a intenção de denotar apenas uso acadêmico, porque mesmo em ambientes comerciais, o aprendizado de máquina geralmente começa experimentalmente.

Ao avaliar diferentes configurações (”hiperparâmetros”) para estimadores ainda há o risco de overfitting no conjunto de teste, porque os parâmetros podem ser ajustados até que o estimador tenha um desempenho ideal. Dessa forma, o conhecimento sobre o conjunto de teste pode “vazar” para o modelo e as métricas de avaliação não informam mais sobre o desempenho de generalização. Para resolver este problema, outra parte do conjunto de dados pode ser realizada como um chamado ”conjunto de validação”: o treinamento continua no conjunto de treinamento, após a avaliação feita no conjunto de validação, e quando o experimento parece ser bem sucedido, a avaliação final pode ser feita no conjunto de teste.

No entanto, ao particionar os dados disponíveis em três conjuntos, reduzimos drasticamente o número de amostras que podem ser usadas para aprender o modelo, e os resultados podem depender de uma escolha aleatória particular para o par de conjuntos de treinamento e validação.

Uma solução para este problema é um procedimento denominado validação cruzada (abreviatura de CV). Um conjunto de teste ainda deve ser apresentado para avaliação final, mas o conjunto de validação não é mais necessário ao fazer o CV. Na abordagem básica, chamada k-fold CV, o conjunto de treinamento é dividido em k conjuntos menores (outras abordagens são descritas abaixo, mas geralmente seguem os mesmos princípios). O seguinte procedimento é seguido para cada uma dos k conjuntos (*”folds”*):

- Um modelo é treinado usando $k - 1$ folds como conjunto de treinamento;
- O modelo resultante é validado na parte restante dos dados (ou seja, é usado como um conjunto de teste para calcular uma medida de desempenho, como acurácia, erro quadrático médio, AUC, etc).

A medida de desempenho relatada pela validação cruzada k-fold é, então, a média dos valores calculados no loop. Esta abordagem pode ser computacionalmente cara, mas não desperdiça muitos dados (como é o caso ao corrigir um conjunto de validação arbitrário), o que é uma grande vantagem em problemas como inferência inversa, onde o número de amostras é muito pequeno.

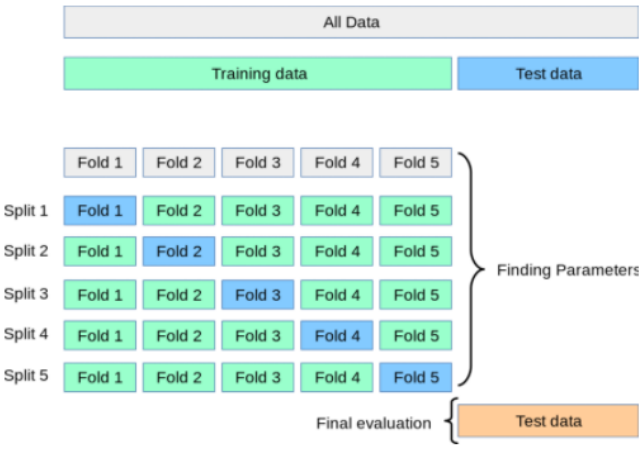

Figura 2. 5 folds CV [19]

Para este trabalho seão utilizadas as métricas AUC, para os problemas de classificação e e o coeficiente de determinação R^2 para os probelmas de regressão, que serão explicados nas subceções a seguir.

A. Curva ROC e Área sob a Curva (AUC)

Uma curva**ROC** [20] é construída traçando a taxa de verdadeiro positivo (TPR) contra a taxa de falso positivo (FPR). A taxa positiva verdadeira é a proporção de observações que foram corretamente previstas como positivas de todas as observações positivas ($TP/(TP+FN)$). Da mesma forma, a taxa de falsos positivos é a proporção de observações que são incorretamente previstas como positivas de todas as observações negativas ($FP/(TN+FP)$). Por exemplo, em testes médicos, a verdadeira taxa positiva é a taxa em que as pessoas são corretamente identificadas para teste positivo para a doença em questão. Sendo:

- TP: Verdadeiro positivo (quantidade de amostras classificadas como positivas sendo positivas);
- TN: Verdadeiro negativo (quantidade de amostras classificadas como negativas sendo negativas);
- FP: Falso positivo (quantidade de amostras classificadas como positivas sendo negativas);
- FN: Falso negativo (quantidade de amostras classificadas como negativas sendo positivas);

Um classificador discreto que retorna apenas a classe prevista fornece um único ponto no espaço ROC. Mas para classificadores probabilísticos, que fornecem uma probabilidade ou pontuação que reflete o grau em que uma instância pertence a uma classe em vez de outra, podemos criar uma curva variando o *threshold* para a pontuação. A Figura 3, exemplifica uma curva ROC.

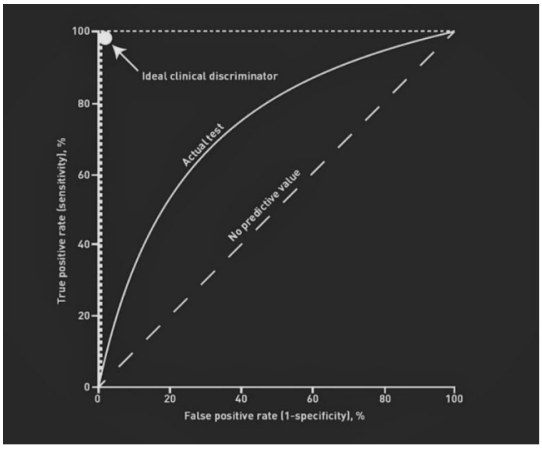

Figura 3. Exemplo de curva ROC

A curva ROC mostra o trade-off entre sensibilidade (ou TPR) e especificidade ($1-FPR$). Classificadores que fornecem curvas mais próximas do canto superior esquerdo indicam um melhor desempenho. Como linha de base, espera-se que um classificador aleatório forneça pontos situados ao longo da diagonal ($FPR = TPR$). Quanto mais perto a curva chega da diagonal de 45 graus do espaço ROC, menos preciso é o teste.

Observe que o ROC não depende da distribuição da classe. Isso o torna útil para avaliar classificadores que prevêem eventos raros (dados desbalanceados), como doenças ou desastres. Em contraste, avaliar o desempenho usando acurácia $(TP+TN)/(TP+TN+FN+FP)$ favoreceria classificadores que sempre predizem um resultado negativo para eventos raros.

Para comparar diferentes classificadores, pode ser útil resumir o desempenho de cada classificador em uma única medida. Uma abordagem comum é calcular a área sob a curva ROC, que é abreviada para **AUC** [21]. É equivalente à probabilidade de que uma instância positiva escolhida aleatoriamente seja classificada mais alta do que uma instância negativa escolhida aleatoriamente, ou seja, é equivalente à estatística de soma de classificação de Wilcoxon de duas amostras [22].

Um classificador com **AUC** alta pode ocasionalmente ter uma pontuação pior em uma região específica do que outro classificador com AUC mais baixa. Mas, na prática, o AUC tem um bom desempenho como uma medida geral de precisão preditiva.

B. coeficiente de determinação R^2

O coeficiente de determinação, também chamado de pontuação R2, é usado para avaliar o desempenho de um modelo de regressão linear. É a quantidade de variação no atributo dependente de saída que é previsível a partir das variáveis independentes de entrada. É utilizado para verificar como resultados bem observados são reproduzidos pelo modelo, dependendo da razão de desvio total dos resultados descrita pelo modelo [23].

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{11}$$

Sendo,

- SS_{res} a soma dos quadrados dos resíduos, também chamada de soma dos quadrados residuais:

$$SS_{res} = \sum_i (y_i - \hat{y}_i)^2 = \sum_i e_i^2 \tag{12}$$

- SS_{tot} a soma total dos quadrados (proporcional à variância dos dados):

$$SS_{res} = \sum_i (y_i - \bar{y_i}) \tag{13}$$

$$\bar{y_i} = \frac{1}{n} \sum_i^n y_i \tag{14}$$

Na melhor das hipóteses, os valores modelados correspondem exatamente aos valores observados, o que resulta em $SS_{res} = 0$ e $R^2 = 1$.. Um modelo base, que sempre prevê a média dos dados observados $\bar{y}$, terá $R^2 = 0$. Modelos com previsões piores do que o modelo base terão um R^2 negativo.

IV. Experimentos e Resultados

Experimentos foram performados com 8 conjunto de dados, sendo 6 do repositório da UCI [2], um conjunto de dados de expressões gênicas [24] e um conjunto de dados de preços de diamantes e vários atributos [25].

Os algoritmos de treinamento dos modelos ELM, RBF e ELm foram implementados. Para o Perceptron e Adaline foi utilizado os pacotes disponibilizados pela biblioteca em Python (scikit-learn) [13].

Todos os conjuntos de dados foram normalizados com média $\bar{x} = 0$ e desvio padrão $\sigma = 1$, removendo amostras com valores ausentes. Para a base de dados de Golub [24] especificamente, foi utilizado o algoritmo PCA [26] para selecionar as características que correspondem por 90% da variância, reduzindo de 7129 expressões gênicas para 22. Como para cada conjunto de dados há hiperparâmetros a serem definitos para cada modelo de rede neural considerado neste trabalho, foi executado uma busca em grid avaliando o resultado da validação cruzada com 10 *folds* avaliando a performance de cada combinação de hiperparâmetros. A busca em grid foi feita variândo o número de neurônios na camada intermediária (p) e o fator de regularização (), para cada um dos algoritmos quando cabivel. Os valores de neurônios na camada intermediária avaliados foram $p = [2, 4, 8, 16, 32, 64, 128, 256, 512, 1024]$, e os valores de fatores de regularização avaliados foram $\lambda = [0, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000, 10000]$. Assim, os melhores hiperparâmetros econtrados, após busca em grid executando validação cruzada com 10 *folds*, se encontram na Tabela I.

Os resultados obtidos para cada conjunto de dados de **classificação** estão listado na Tabela II, onde N e N_d correspondem à quantidade total de amostras e atributos, respectivamente, N^+ é o número de amostras positivas e N^- o número de amostras negativas. As performances médias (AUC) [3] e desvios padrões após dez ensaios, foram calculadas para conjuntos de testes separados (2). Os resultados obtidos indicam que apenas os modelos ELm e Perceptron apresentaram resultados satisfatórios, estando condizentes com os resultados obtidos em: [27] e com resultados de vários trabalhos presentes no Kaggle (https://www.kaggle.com/), comunidade de cientistas de dados e praticantes de aprendizado de máquina, já que são conjuntos de dados amplamente utilizados como *benchmark*.

Os resultados obtidos para cada conjunto de dados de **regressão** estão listados na Tabela III, onde N e N_d correspondem à quantidade total de amostras e atributos. As performances médias (Coeficiente R^2) [23] e desvios padrões após dez ensaios, foram calculadas para conjuntos de testes separados (2). Os resultados obtidos indicam que apenas os modelos ELm e Adaline apresentaram resultados satisfatórios, estando condizentes com os resultados obtidos em vários trabalhos presentes no Kaggle (https://www.kaggle.com/), comunidade de cientistas de dados e praticantes de aprendizado de máquina, já que são conjuntos de dados amplamente utilizados como *benchmark*.

Acredito que se fosse utilizado o algoritmo PCA para escolher apenas os padrões mais adequados para serem aprendidos, o grau no qual a solução Hebbiana difere da solução de erro zero poderia ser mais controlado, obtendo assim métricas de performance melhores. Isso tem um efeito de penalização em relação ao resultado da pseudoinversa, o que se espera que resulte em uma curva de aprendizado mais suave [1].

V. Análise de Conclusão

A partir desse trabalho pode-se observar a importância na seleção de hiperparâmetros nos modelos, pois impacta bastante em seus desempenhos, e como a escolha dos melhores hiperparâmetros varia bastante em função da base de dados. Entretanto, deve-se levar em consideração que a busca em grid realizando validação cruzada para seleção de modelos é um processo muito custoso, dependedo da quantidade de valores dos hiperparâmetros a serem avaliados.

A partir dos resultados obtidos observa-se uma certa deficiência no algoritmo RBF com centros e raios determinados pelo algoritmo k-médias, pois os resultados obtidos não foram satisfatórios. Os resultados obtidos com o ELM Hebbiano também não foram satisfatórios, apresentando boas métricas em apenas alguns conjuntos de dados. Conforme o trabalho do Euler [1], se fosse adotado alguma técnica para escolher os padrões mais representativos, poderiam ser obtidos resultados melhores. Os resultados obtidos com o ELM, Perceptron e Adaline foram satisfatórios, estando próximos dos resultados obtidos em [27] e com resultados de vários trabalhos presentes no Kaggle (https://www.kaggle.com/).

Referências

[1] E. G. Horta, "Aplicação de máquinas de aprendizado extremo ao problema de aprendizado ativo," 2015.

[2] A. Asuncion and D. Newman, "Uci machine learning repository," 2007.

[3] C. Ferri, J. Hernández-Orallo, and R. Modroiu, "An experimental comparison of performance measures for classification," *Pattern Recognition Letters*, vol. 30, no. 1, pp. 27–38, 2009.

[4] J. A. Colton and K. M. Bower, "Some misconceptions about r2," *International Society of Six Sigma Professionals, EXTRA-Ordinary Sense*, vol. 3, no. 2, pp. 20–22, 2002.

[5] D. Krstajic, L. J. Buturovic, D. E. Leahy, and S. Thomas, "Cross-validation pitfalls when selecting and assessing regression and classification models," *Journal of cheminformatics*, vol. 6, no. 1, pp. 1–15, 2014.

[6] D. O. Hebb, *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

[7] H. Anton and R. C. Busby, *Contemporary linear algebra*. Wiley, 2003.

[8] J. A. Hertz, *Introduction to the theory of neural computation*. CRC Press, 2018.

[9] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[10] W. Pitts and W. S. McCulloch, "How we know universals the perception of auditory and visual forms," *The Bulletin of mathematical biophysics*, vol. 9, no. 3, pp. 127–147, 1947.

[11] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological review*, vol. 65, no. 6, pp. 386–408, 1958.

[12] B. Widrow and M. E. Hoff, "Adaptive switching circuits," tech. rep., Stanford Univ Ca Stanford Electronics Labs, 1960.

[13] "scikit-learn: machine learning in python — scikit-learn 0.24.1 documentation."

[14] W. F. Schmidt, M. A. Kraaijveld, R. P. Duin, *et al.*, "Feed forward neural networks with random weights," in *International Conference on Pattern Recognition*, pp. 1–1, IEEE COMPUTER SOCIETY PRESS, 1992.

Tabela I
MELHORES HIPERPARÂMETROS ECONTRADOS APÓS BUSCA EM GRID EXECUTANDO VALIDAÇÃO CRUZADA COM 10 *FOLDS*

	Perceptron	ELM		RBF		ELM Hebbiano	Adaline
Conjunto de Dados	λ	p	λ	p	λ	p	λ
Ionosphere	0	32	0.001	16	100	32	-
EEG Eye State	0	1024	0.0001	64	10000	16	-
Golub	0.01	128	1000	2	1	64	-
Fertility Diagnosis	0.0001	16	0.0001	2	0	64	-
Indian Liver Patient	0.01	128	1	64	10000	64	-
Banknote Authentication	0	64	0.0001	64	1000	64	-
Concrete	-	1024	1	32	10	512	0.001
Diamonds	-	512	0.001	64	10	512	0.0001

Tabela II
VALORES MÉDIOS DE AUC E CARACTERÍSTICAS DOS CONJUNTOS DE DADOS

	Perceptron	ELM	RBF	ELM Hebbiano	N_d	N	N^+	N^-
Conjunto de Dados								
Ionosphere	$\mathbf{0.861 \pm 0.000}$	0.849 ± 0.032	0.500 ± 0.000	0.704 ± 0.044	33	351	225	126
EEG Eye State	0.538 ± 0.000	$\mathbf{0.860 \pm 0.004}$	0.538 ± 0.004	0.546 ± 0.011	14	14980	6723	8257
Golub	$\mathbf{0.900 \pm 0.000}$	0.882 ± 0.037	0.570 ± 0.136	0.731 ± 0.064	7129	72	25	47
Fertility Diagnosis	$\mathbf{0.728 \pm 0.000}$	0.528 ± 0.066	0.500 ± 0.000	0.512 ± 0.022	9	100	12	81
Indian Liver Patient	$\mathbf{0.702 \pm 0.000}$	0.545 ± 0.013	0.537 ± 0.025	0.532 ± 0.034	10	579	414	165
Banknote Authentication	0.985 ± 0.000	$\mathbf{1.000 \pm 0.000}$	0.970 ± 0.013	0.836 ± 0.030	4	1372	610	762

Tabela III
VALORES MÉDIOS DO COEFICIENTE R^2 E DIMENSÕES DOS CONJUNTOS DE DADOS

	Adaline	ELM	RBF	ELM Hebbiano	Nd	N
Conjunto de Dados						
Concrete	0.569 ± 0.091	0.901 ± 0.004	0.394 ± 0.020	-3.700 ± 0.583	8	1030
Diamonds	0.920 ± 0.000	0.964 ± 0.001	0.467 ± 0.019	-0.972 ± 0.000	23	53940

[15] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[16] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern recognition*, vol. 36, no. 2, pp. 451–461, 2003.

[17] M. JL, "Orr,"introduction to radial basis function networks."(1996)," 1996.

[18] M. M. RAMADHAN, I. S. SITANGGANG, F. R. NASUTION, and A. GHIFARI, "Parameter tuning in random forest based on grid search method for gender classification based on voice frequency," *DEStech Transactions on Computer Science and Engineering*, no. cece, 2017.

[19] "Cross-validation: evaluating estimator performance."

[20] J. Fan, S. Upadhye, and A. Worster, "Understanding receiver operating characteristic (roc) curves," *Canadian Journal of Emergency Medicine*, vol. 8, no. 1, pp. 19–20, 2006.

[21] C. X. Ling, J. Huang, H. Zhang, *et al.*, "Auc: a statistically consistent and more discriminating measure than accuracy," in *Ijcai*, vol. 3, pp. 519–524, 2003.

[22] P. D. Bridge and S. S. Sawilowsky, "Increasing physicians' awareness of the impact of statistics on research outcomes: comparative power of the t-test and wilcoxon rank-sum test in small samples applied research," *Journal of clinical epidemiology*, vol. 52, no. 3, pp. 229–235, 1999.

[23] "Coefficient of determination - wikipedia."

[24] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, *et al.*, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *science*, vol. 286, no. 5439, pp. 531–537, 1999.

[25] "Your machine learning and data science community."

[26] S. Roweis, "Em algorithms for pca and spca," *Advances in neural information processing systems*, pp. 626–632, 1998.

[27] L. C. Torres, C. L. Castro, F. Coelho, and A. P. Braga, "Large margin gaussian mixture classifier with a gabriel graph geometric representation of data set structure," *IEEE transactions on neural networks and learning systems*, 2020.