Universidade Federal do Ceará

Disciplina: Inteligência Computacional Aplicada
Professor(a): Guilherme Barreto
Estudante: Rubem Vasceconcelos Pacelli Matrícula: 472525

Junho/2022

Neural network - Report

1 Work 01 - Rosenblatt's perceptron

This work considers a classification problem for a multivariate dataset. The iris flower dataset is utilized in a Rosenblatt's perceptron to classify one class from the others. The supervised learning utilizes a common procedure that comprises data preparation, training, test, and analysis of the accuracies for N_r independent realizations.

The Rosenblatt's perceptron consist in the neuron model architecture, introduced by McCulloch and Pitts in 1943, with a learning algorithm that adjusts the synaptic weights in a supervised fashion. The McCulloch and Pitts' activation function is a step function that triggers the output from 0 to 1 when the induced local field overpass the threshold. This method is effective for binary classification of linearly separable problems, where one can sketch a straight line that divides the classes without overlapping.

At the instant n, the induced local field is given by

$$v(n) = \mathbf{w}^{\mathsf{T}}(n)\, \mathbf{x}(n), \tag{1}$$

where $\mathbf{w}^{\mathsf{T}}(n) = \begin{bmatrix} w_0(n) & w_1(n) & \cdots & w_{N_a}(n) \end{bmatrix}^{\mathsf{T}}$ and $\mathbf{x}^{\mathsf{T}}(n) = \begin{bmatrix} x_0(n) & x_1(n) & \cdots & x_{N_a}(n) \end{bmatrix}^{\mathsf{T}}$ are the synaptic weights of the perceptron and the input signal, respectively, and N_a indicated the number of attributes. The elements $w_0(n)$ and $x_0(n) \triangleq +1$[1] are, respectively, the bias and its input.

the machine learning algorithms settles on the well-established theory of adaptive filters. Particularly for the Rosenblatt's perceptron, it is utilized the Least-Mean-Square (LMS) algorithm, that aims to make a instantaneous approximation of the gradient vector. The optimization algorithm is given by

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta \mathbf{g}(n), \tag{2}$$

where η is the step-learning hyperparameter and $\mathbf{g}(n) \triangleq \nabla \mathscr{E}(\mathbf{w})$ is the stochastic approximation of the gradient vector, being $\mathscr{E}(\mathbf{w})$ the cost function. The 1 pass through the step function, $\varphi(\cdot)$, generating the perceptron output, $y(n) = \varphi(v(n)) \in \{0, 1\}$.

[1] Depending on the author, it can the defined as -1.

This signal is compared to the desired value, $d(n) \in \{0, 1\}$, generating the error signal $e(n) = d(n) - y(n) \in \{-1, 0, 1\}$, which indicates whether the perceptron misclassified or not. The LMS algorithm uses instantaneous value of the MSE (Mean-Squared Error) cost function, that is,

$$\mathscr{E}(\mathbf{w}) = \frac{1}{2}e^2(n). \tag{3}$$

Differentiating this equation with respect to the synaptic weights, we get

$$\mathbf{g}(n) = \frac{\partial \mathscr{E}(\mathbf{w})}{\partial \mathbf{w}(n)} = -\mathbf{x}(n)e(n). \tag{4}$$

Substituting (5) into (2), it yields the learning equation, given by

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta\mathbf{x}(n)e(n). \tag{5}$$

The Algorithm 1 summarizes the procedure utilized for the Rosenblatt's perceptron, including data preparation techniques, such as hand-out technique and shuffling the dataset. The method utilized $N_r = 20$ independent realization, and pass through the training set $N_e = 100$ epochs. At the end of each realization, it is stored the accuracy reached by the test data, and the results of all accuracies are investigated in terms of mean and standard deviation. The iris dataset contains $N = 150$ instances with $N_a = 4$ attributes (petal length, petal width, sepal length, and sepal width) and $K = 3$ classes (Setosa, Versicolour, and Virginica). It was chosen a ratio of $80\% - 20\%$ for the training and testing dataset, respectively.

Algorithm 1: Rosenblatt's perceptron

```
Input: $\mathbf{X}, \mathbf{D}$                                  /* attributes and labels dataset */
forall $\{1, 2, \cdots, N_r\}$ do
    $\mathbf{w}(n) \leftarrow$ initialize
    $\mathbf{X}, \mathbf{D} \leftarrow$ shuffle
    $(\mathbf{X}_{trn}, \mathbf{D}_{trn}), (\mathbf{X}_{tst}, \mathbf{D}_{tst}) \leftarrow$ hold-out   /* training and testing dataset */
    forall $\{1, 2, \cdots, N_e\}$ in the training dataset do
        $v(n) \leftarrow \mathbf{w}^\mathsf{T}(n)\,\mathbf{x}(n)$
        $y(n) \leftarrow \varphi(v(n))$
        $\mathbf{w}(n+1) \leftarrow \mathbf{w}(n) + \eta\mathbf{x}(n)e(n)$
        $\mathbf{X}_{trn}, \mathbf{D}_{trn} \leftarrow$ shuffle
    $accuracy \leftarrow$ test$(\mathbf{X}_{tst}, \mathbf{D}_{tst})$
```

The process described in Algorithm 1 was repeated for the classification problem of each class. the Table 1 shows the results for each class. The setosa classes clearly outperforms the other classes since it is linearly separable for some attributes, as shown

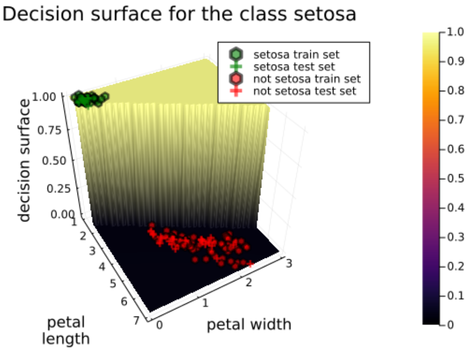

Figure 1: Decision surface of setosa class.

Table 1: Rosenblatt's perceptron performance for classification problem

Classes	mean accuracy	standard deviation
Setosa	98.33	0.01972
Virginica	54.16	0.1251
Versicolor	53.66	0.1591

in the decision surface in Figure 1[2].

The confusion matrix of the setosa class is shown in Figure 2 for the first realization. The main diagonal indicates that there were not false negative or false positive.

The Figure 3 shows the evolution of the training dataset accuracy throughout the epochs. One can notice the fast convergence to accuracy of 100%.

[2] Since the problem has four attributes, this plot would be impossible as the would get 2 degree of freedom. Therefore, for this result, we considered only these two attributes.

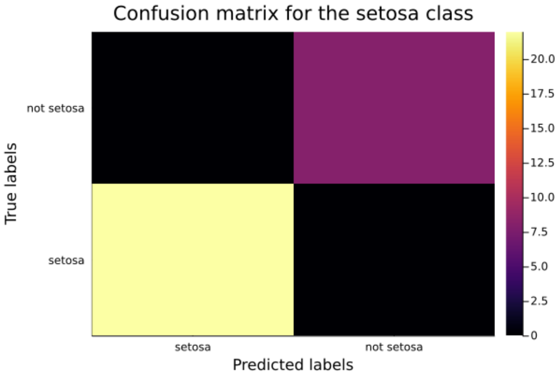

Figure 2: confusion matrix for setosa class.

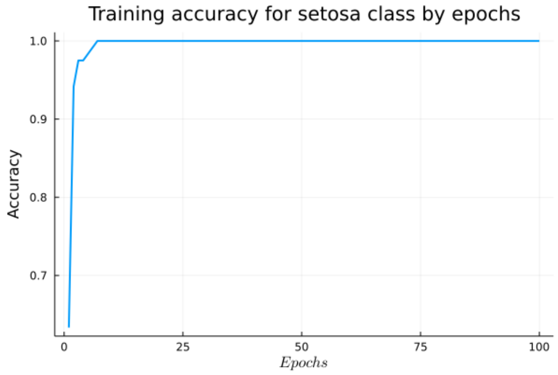

Figure 3: Training dataset evolution for the setosa classification.

For a dummy dataset whose the $K = 4$ classes, the Rosenblatt's perceptron achieved a mean accuracy of 97.5% and a standard deviation of 0.05. The Figure 4 shows the decision surface of the desired class for the realization whose accuracy the closest to the mean accuracy. All instances of all classes are samples drawn from a Gaussian distribution with a given mean and variance.

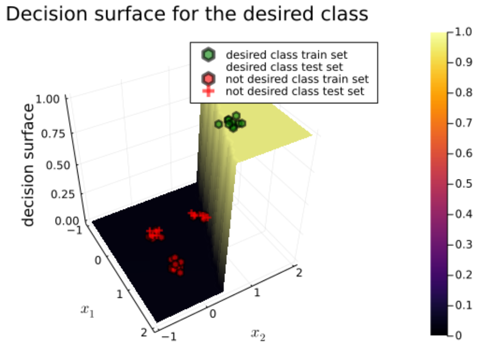

Figure 4: Decision surface for the desired class.

2 ADALINE

The Adaptive Linear Element (or ADALINE) is a variation of the Rosenblatt's perceptron, where the step function is replaced by a linear function, that is, $y(n) = \varphi(u(n)) = u(u)$. One can combine a tapped delay line with an ADALINE, thus creating an adaptive filter.

Consider a regression problem of adaptive filtering where the input signal is the desired signal corrupted by a Gaussian noise. The ADALINE model tries to retrieved the original data using the same process described in Algorithm 1. However, the performance analysis is around the MSE error instead the accuracy since it is now a regression problem.

The Table 2 shows the performance of the mean MSE and its standard obtained over the independent realizations, in addition the root mean squared error (RMSE).

Table 2: ADALINE performance for regression problem

$d(n)$	MSE mean	standard deviation
$ax + b$	98.33	0.01972
$ax^2 + bx + c$	54.16	0.1251