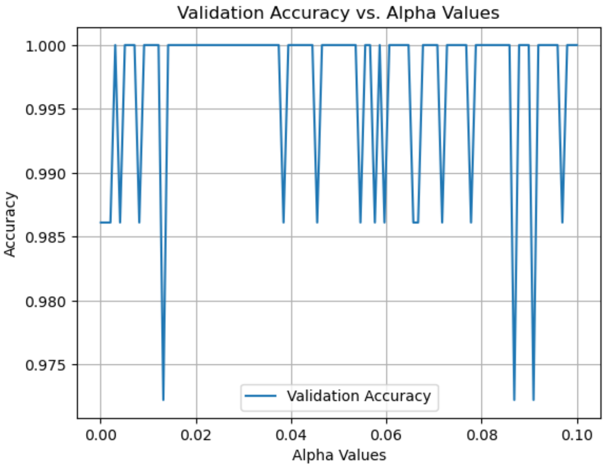
Project 2: The Perceptron

Team RosAlene: Rosato Sara, Khachmadi Arsalene

The Perceptron is a supervised learning binary classifier algorithm that is based on the idea of the existence of a hyperplane $\mathcal{H} = \{x : w^T x + \hat{b} = 0\}$ which separates the data (decision boundary); where w is the weight vector, while b is the bias. Both are initialized in our model by the function `initialize_weights`. To classify a new point, we evaluate where this is with respect to the hyperplane.
The error function used is the perceptron criterion $E_p(w) = -\sum_{i \in \mathcal{M}} w^T x_i y_i$, which tries to minimize the quantity $w^T x_i$ using the stochastic gradient descent (SGB) as an update step for the weights (and bias) each time we had a misclassified point ($\in \mathcal{M}$).
After initializing the model's weights, bias, and learning rate (which is given), the core method is the train function, which takes the matrix X and the vector of labels y associated with that matrix as inputs. For each sample of each iteration, it makes a prediction and checks whether the point is correctly classified or not, updating the weights and bias with the function `gradient_descent_step`.
To do some experiments, we loaded the "digits" dataset, using only the first two classes (0 and 1). Since the Perceptron assumes that data are linearly separable and that is possible to do binary classification $\{y_i \in \{-1, 1\}\}$, we transformed all the y features equals to 0 in -1. After splitting our dataset in train (60%), test (20%), and validation (20%), we performed a hyperparameter tuning for the learning rate α, taking the best one based on the model's accuracy on the validation set. We obtained $\alpha = 0.003127$ with an accuracy of 1.

We then used the best value of α to train our model using the training set, the validation set to choose the model and, at the end, we evaluated it using the testing set. The accuracy obtained is equal to 0.9861. We evaluated our model also using precision [1], recall [2], and f1-score [3] obtaining satisfying results. The model, in fact, seems to perform well on the testing set, making correct predictions while minimizing both false positives and false negatives.
To be sure of the correctness of our code, we compared it against the scikit-learn implementation of the Perceptron and we saw that the accuracy on the validation set is the same.



0.1 ChatGPT Policy

Used to have a reference for the plot.

[1] number positive predictions well made

[2] percentage of positives well predicted by our model

[3] evaluation of the performance of the model, harmonic mean of precision and recall