PERCEPTRON

# ABSTRACT

In this paper, we will study one of the pillars of today’s Artificial Intelligence algorithms: the Perceptron Learning Algorithm (PLA). This algorithm is a basic system that allows a linear separation of a linearly separable dataset. It can be used for basic problems where a binary separation can be done. Our small team of 4 members worked on this subject for 1 month, learning the theory behind it and testing it using Python code.

We will first talk about the origins of the PLA, how it was discovered, and in what context. Then, we will theorize the concept using mathematical expressions, and finally, provide a working example using Python.

# HISTORY

# THEORY

The PLA is a simple algorithm. It computes a scalar product between a vector , to the data vector , with . The result is the corresponding class - or label - of the data. The label is either or , separating data between positive and negative labels. Let be the label of .

When beginning, the algorithm starts with a random equation. The initialization only affects the time that it will take to correctly separate data: if the dataset is linearly separable, the PLA will always find the best vector to correctly classify the whole dataset.

If incorrectly classifies the data, we can correct the value of using:

Here, is the “amount of correction” that we apply to . This parameter is often called the **learning rate** (LR) in modern AI applications. We will later discuss about the impact of different LR values on the amount of iterations needed to the PLA to find the perfect .

We can now create the algorithm, by assembling the bits that we talked about earlier:

This is a very simple algorithm, with 1 condition. If the condition is satisfied ( correctly classifies the data), then nothing changes. Else, we update .

TODO: faire un schema pour bien expliquer comment le vecteur w classifie x. Peut montrer le lien avec un reseau de neuronnes.

# PRACTICE

The practice isn’t very complex. In fact, the algorithm is very basic and needs only a single Python function of a few lines.

A screen shot of a computer

Description automatically generated

TODO: faire une loss function, des métriques, etc pour tester si le modèle classifie bien les données.

# CONCLUSION