TRUMAN
STATE UNIVERSITY

Professional Program in
DATA SCIENCE

PDAT615G: Machine Learning

Module 2 – Perceptrons, Optimization and Support Vector Machines

DISTINCT BY DESIGN

Module 2: Introduction

- Linearly separable sets
- The *perceptron* neuron
  - Side-tour of vector arithmetic
  - Visual explanation
- Methods of optimization and applications to the perceptron
- Support Vector Machines
  - Building on the idea of the perceptron
  - Classification for more than two categories
- Introduction to neural networks

The Perceptron

Linearly Separable Sets

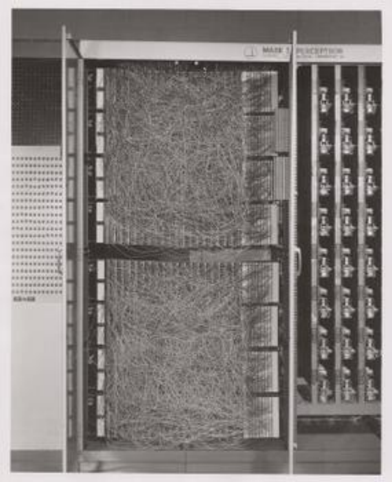
Two sets of points are *linearly separable* when

- in two dimensions, they can be separated by a line,
- in three dimensions, they can be separated by a plane, or
- generally, in n dimensions, they can be separated by an $(n-1)$-dimensional *hyperplane*: the set of points $x_1, \ldots, x_n$ such that

  $$w_1x_1 + w_2x_2 + \cdots + w_nx_n = \text{a constant}.$$

Rosenblatt's *Perceptron* is an algorithm (and device!) to find such separating hyperplanes.

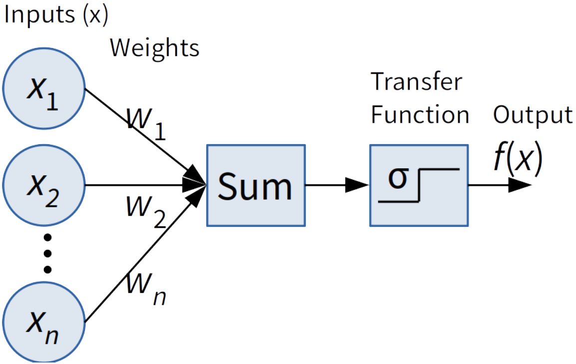
- Developed by Rosenblatt in the 1950's as an artificial "neuron."
- More than one could be stacked to separate more complex regions.
- A conceptual basis that leads to SVM's and neural networks.
- It's abilities were, perhaps, initially over-promised.



The Mark I Perceptron (wikipedia.org)

The perceptron passes a weighted sum of inputs through a transfer function to produce binary output.

Transfer function σ limits output to [0, 1]: $\sigma(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$



A Side-Trip through Linear Algebra

The language of vectors (linear algebra) helps encapsulate the geometry we're dealing with.

- *Points* are locations in Euclidean space.
- *Vectors* represent *directions* and *magnitudes* in Euclidean space.
- Sometimes we blur the difference:
  Points $\approx$ Vectors based at $(0, 0)$

Vector Arithmetic

- Let $\vec{a} = \langle a_1, a_2 \rangle$,
  and $\vec{b} = \langle b_1, b_2 \rangle$.
- Length of $\vec{a}$:
  $|\vec{a}| = \sqrt{a_1^1 + a_2^2}$.
- Sum of $\vec{a}$ and $\vec{b}$:
  $\vec{a} + \vec{b} = \langle a_1 + b_1, a_2 + b_2 \rangle$.

Dot products give a single number that measures how much the directions of two vectors coincide.

Multiply corresponding vector components and add:

$$\vec{a} \cdot \vec{b} = a_1 b_1 + a_2 b_2.$$

Multiply vector lengths and the cosine of the angle θ between the vectors:

$$\vec{a} \cdot \vec{b} = |\vec{a}||\vec{b}| \cos(\theta).$$

Dot Product	Directions of $\vec{a}$ and $\vec{b}$
$\vec{a} \cdot \vec{b} = 0$	Perpendicular
$\vec{a} \cdot \vec{b} > 0$	More in the same direction
$\vec{a} \cdot \vec{b} < 0$	More in the opposite direction

A direction vector has a “dual” hyperplane.

- Direction vector: $\vec{w} = \langle w_1, \ldots, w_n \rangle$.
- “Dual” Plane: All points reached by vectors perpendicular to $\vec{w}$.
  - All $\vec{x}$ such that $\vec{w} \cdot \vec{x} = 0$.
  - All $\langle x_1, \cdots, x_n \rangle$ such that
    $w_1 x_1 + w_2 x_2 + \cdots + w_n x_n = 0$.
- Note, any two $\vec{w}_1$ and $\vec{w}_2$ that are
  in the same direction give the same plane.

Simplest Case: Separating Planes through the Origin

Finding a separating plane through the origin is equivalent to finding direction vector $\vec{w}$ such that

- $\vec{w} \cdot \vec{x} > 0$ for all $\vec{x}$ in the "positive" group.
- $\vec{w} \cdot \vec{x} < 0$ for all $\vec{x}$ in the "negative" group.

Note: Such a solution is not unique.

If groups are separable *not* through the origin, raise all points into the next higher dimension.

- A hyperplane through the origin can now separate the groups.
- This is equivalent to moving the base of the direction vector and plane by a constant vector $\vec{b}$.

Key Idea: The perceptron algorithm pull the direction vector toward any point that’s misclassified.

The Perceptron Algorithm

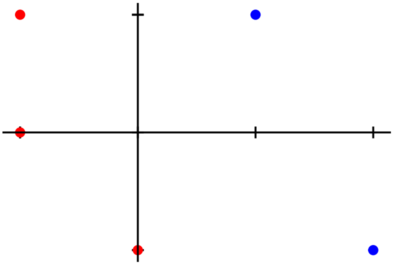
- Start with explanatory variables $x_1, \dots, x_n$ and 0/1 response y.
- Step 0: Add a new column x_0 that's all 1's.
- Step 1: Start with a random weight vector $\vec{w} = \langle w_1, \dots, w_n \rangle$.
- Step 2: (Forward Calculation) For the first data point, calculate $f(\vec{x}) = \sigma(\vec{w} \cdot \vec{x}) = \sigma(w_1 x_1 + \cdots + w_n x_n) = 0$ or 1.
- Step 3: (Back Propagation) If $f(\vec{x})$ and y don't agree, modify $\vec{w}$:

$$\vec{w} \leftarrow \vec{w} + (y - f(\vec{x}))\,\vec{x}.$$

- Step 4: Repeat with the next $\vec{x}$. Algorithm *converges* when all points are correctly classified.

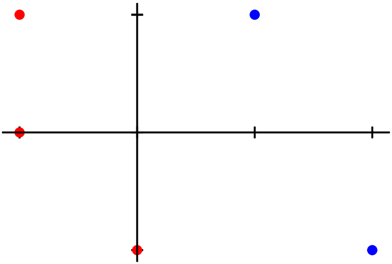
Example: Perceptron in Action!

x1	x2	y
1	1	1
2	-1	1
-1	1	0
-1	0	0
0	-1	0



Example: Perceptron in Action!

x1	x2	y
1	1	1
2	-1	1
-1	1	0
-1	0	0
0	-1	0



On its own, the perceptron has several limitations.

- If the data is *not* linearly separable, the algorithm may not converge, and can instead cycle.
- The solution is not unique.
- The solution depends on the order of the inputs.
- The solution may not be the most general for predicting on new data.

But understanding the perceptron is a good foundation for what comes next, and the name evokes the optimism of the 1950’s!