Perceptrons

CS 510
Lecture #11
March 25, 2020
Start Here: Feature Space

- Samples are whatever you are classifying
  - In this case people
  - In future cases, images
- Every sample has N features
  - In this case, N=2
  - Features are age & height
- Every sample is a point in feature space
Supervised Learning

- Supervised learning assumes a label for every training sample
  - In this case, adult = T/F
  - T = red
- The goal is to divide feature space according to the labels
  - In this case, age >= 18 is good
Supervised Learning II

- Labels are different from samples
- The same data set can have multiple labelings
  - Now I have changed the label to “tall”
  - Tall = red
- Learning maps data to labels
Supervised Learning III

- Single Linear function at times not enough
- Consider the label “tall adults”
  - Now a non-linear separator is needed
  - Note that the red line is right for the training samples
  - But may fail for new test samples
    - Like tall 10 year olds!
Formalism

• Let \( x = [v_1, v_2, \ldots, v_D] \) be a data sample
  – \( D \) is the dimensionality of the feature space
  – So \( x \in \mathbb{R}^D \)

• Let \( X = \{x_1, x_2, \ldots, x_N\} \) be a training set
  – \( N \) is the number of training samples

• Let \( Y = \{y_1, y_2, \ldots, y_N\} \) be a label set
  – \( Y \)'s are scalars, not vectors
  – \( Y \)'s may be in \( \{0,1\} \) or a discrete label set
  – The \( N \)'s match

• Goal: learn \( f() \) such that \( y_i = f(x_i) \)
Error Functions

• In general, no such f() exists
  – For example, if \( x_i = x_j \), but \( y_i \neq y_j \)
  – How could this happen?
    • Noise in data features
    • Noise in label set
    • Probabilistic concept

• So instead, minimize an error function
  – e.g. \( Err = \sum_i (y_i - f(x_i))^2 \)
Simple Example

• Data from previous slide
  Features are age & height, so $D = 2$
  11 samples, so $N = 11$
  Label concept is “tall adult”

• $X = \{[48, 4.5], [47, 6], \ldots, [3,2]\}$

• $Y = \{0, 1, \ldots, 0\}$

• Find $f(x)$ that minimizes the squared error
A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY

WARREN S. McCulloch AND WALTER PITTS

FROM THE UNIVERSITY OF ILLINOIS, COLLEGE OF MEDICINE,
DEPARTMENT OF PSYCHIATRY AT THE ILLINOIS NEUROPSYCHIATRIC INSTITUTE,
AND THE UNIVERSITY OF CHICAGO

Because of the “all-or-none” character of nervous activity, neural
events and the relations among them can be treated by means of proposi-
tional logic. It is found that the behavior of every net can be described
in these terms, with the addition of more complicated logical means for
nets containing circles; and that for any logical expression satisfying
certain conditions, one can find a net behaving in the fashion it describes.
It is shown that many particular choices among possible neurophysiological
assumptions are equivalent, in the sense that for every net behaving
under one assumption, there exists another net which behaves under
the other and gives the same results, although perhaps not in the
same time. Various applications of the calculus are discussed.

I. Introduction

Theoretical neurophysiology rests on certain cardinal assump-
tions. The nervous system is a net of neurons, each having a soma
and an axon. Their adjunctions, or synapses, are always between the

Arguably the start of modeling mathematically the behaviors exhibited by networks of neurons.
CORNELL AERONAUTICAL LABORATORY, INC.
BUFFALO, N. Y.

REPORT NO. 85-460-1

THE PERCEPTRON
A PERCEIVING AND RECOGNIZING AUTOMATON
(PROJECT PARA)

January, 1957

Prepared by: Frank Rosenblatt,
Project Engineer
Perceptrons

• Technique:
  – Find a hyperplane that separates true samples (y = 1) from false samples (y = 0)

• Formula:
  – \( f(x) = h(w \cdot x + b) \)
  – \( w \) and \( b \) are learned weights
  – \( h(z) = 1 \) if \( z > 0 \), otherwise \( h(z) = 0 \)
  – The hyperplane geometry should be clear…
Training a Perceptron

- Initialize all $w$'s & $b$ to small random values
- For $iter = 1$ to count do
  - For sample $i = 1$ to $N$ do
    - $d_i(iter) = h(w(iter - 1) \cdot x + b(iter - 1))$
    - For weight $j = 1$ to $D$
      - $w_j(iter) = w_j(iter - 1) + (y_i - d_i(iter))x_i[j]$
    - $b(iter) = b(iter - 1) + (y_i - d_i(iter))$
0 = wx+b is a line in feature space

- Initially random
- Line shown:
  - W1 = 0.25
  - W2 = 0.21
  - B = -5

Remember, y is the output (not subject height)
What happens when we look at sample x?

x is true, so y = 1.
x is on the positive side of the line, so d=1
So y-d is 0, and nothing changes
Now what happens when we look at a new sample $x$?

- $x$ is false, so $y = 0$.
- $x$ is on the negative side of the line, so $d=1$

So $y-d$ is $-1$

- $B$ gets smaller
  - Pushing the line toward the sample
- $W_1$ gets smaller
  - 25x more than $b$
- $W_2$ gets smaller
  - 60x more than $b$
This picture from wikipedia tries to show how the line adjusts with each sample.
Convergence

• Does this algorithm converge?
  – Yes, if the data is linearly separable
  – But not uniquely: the decision boundary will fall somewhere in the gap between the data
Excellent Online Visualizations

An Interactive Journey into Machine Learning
Convergence II

• Does this algorithm converge?
  – No, if the data is not separable
  – There are variants that converge
    • If \( y \in \{0, 1\} \), variations will converge to maximize the number of correctly labeled samples
Neural Network Interpretation

[Diagram of a neural network with input, hidden, and output layers, with input nodes labeled Input #1, #2, #3, #4, and connections to hidden nodes, which then connect to the output node labeled Output.]
The Critic(s)

Minsky and Papert literally wrote the textbook. In so doing kicked off a bit of a fire storm – in the process throwing a large amount of cold water on the whole neural network idea (and promise).
For Example – Solve This!

The XOR problem

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>0</td>
</tr>
</tbody>
</table>
More Generally

• Limited to a single, linear decision boundary....
Multi-layer Perceptrons

- Can we combine perceptrons to learn more complex decision boundaries?

\[ h(w \cdot x + b) \]

\[ [o_1, o_2] \]

\[ h(w \cdot x + b) \]

\[ h(w \cdot x + b) \]

\[ [v_1, v_2, \ldots, v_D] \]
A Rebirth, one of several
Other Linear Classifiers...

- Gaussian Models
  - Model classes by
    - Means $\mu_0$, $\mu_1$
    - Covariance $\Sigma$ (one matrix, not two)
    - Measure distances to means in standard deviations
    - Select closest mean
  - Decision boundary will be linear
Other Linear Classifiers...

• Support Vector Machines (SVMs)
  – Find the line that maximizes the margin (gap) between the decision boundary & samples
  – Note: we will (hopefully) discuss kernels later
Multi-layer Perceptrons II

• Classic perceptrons threshold linear functions
  \[ f(x) = h(w \cdot x + b) \]
  – \( h() \) is a threshold-based activation function
  – Converts activations into decisions

• But if we want to combine perceptrons?
  – Thresholding individual perceptrons is not useful
  – Replacing \( h() \) with identity would allow us to sum linear responses
  – But a sum of linear responses is just another linear response
Sigmoid Activation Functions

- \( f(x) = s(w \cdot x + b) \)

\[ y = \tanh(x) \]
\[ y = (1 + e^{-x})^{-1} \]
Activation Function Properties

• Activation functions must
  – Be non-linear

• Activation functions may
  – map an infinite domain to a finite range
    • Like [-1, 1] (for tanh) or [0, 1] (for logistic)
    • Keeps values from growing too large/small
    • Sometimes called “squashing”
  – Have non-zero derivatives everywhere
    • Useful for training