Machine Learning is…

Machine learning, a branch of artificial intelligence, concerns the construction and study of systems that can learn from data.

Machine Learning is…

Machine learning is programming computers to optimize a performance criterion using example data or past experience.

-- Ethem Alpaydin

The goal of machine learning is to develop methods that can automatically detect patterns in data, and then to use the uncovered patterns to predict future data or other outcomes of interest.

-- Kevin P. Murphy

The field of pattern recognition is concerned with the automatic discovery of regularities in data through the use of computer algorithms and with the use of these regularities to take actions.

-- Christopher M. Bishop
Machine Learning is...

Machine learning is about predicting the future based on the past.

-- Hal Daume III

Machine Learning, aka

data mining: machine learning applied to "databases", i.e. collections of data

inference and/or estimation in statistics

pattern recognition in engineering

signal processing in electrical engineering

induction

optimization
3/5/15

Data

Data examples

IMDb
IMDb
IMDb
IMDb

Data

examples

IMDb
IMDb
IMDb
IMDb

Supervised learning

Supervised learning: given labeled examples

Examples

label
label_1
label_3
label_4
label_5

labeled examples
Supervised learning: given labeled examples

Supervised learning: learn to predict new example

Supervised learning: classification

Classification Example

Differentiate between low-risk and high-risk customers from their **income** and **savings**

*Classification: a finite set of labels*
Classification Applications

- Face recognition
- Character recognition
- Spam detection

Biometrics: Recognition/authentication using physical and/or behavioral characteristics: Face, iris, signature, etc

...

Supervised learning: regression

<table>
<thead>
<tr>
<th>label</th>
<th>-4.5</th>
<th>10.1</th>
<th>3.2</th>
<th>4.3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Regression: label is real-valued

Regression Applications

- Economics/Finance: predict the value of a stock
- Epidemiology
- Car/plane navigation: angle of the steering wheel, acceleration, …
- Temporal trends: weather over time

Regression Example

Price of a used car

\[ y = wx + w_0 \]

\( x \): car attributes (e.g. mileage)
\( y \): price

...
Unsupervised learning

Unsupervised learning: given data, i.e. examples, but no labels

Unsupervised learning applications

- Learn clusters/groups without any label
- Customer segmentation (i.e. grouping)
- Image compression
- Bioinformatics: learn motifs

Reinforcement learning

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Reward</th>
</tr>
</thead>
<tbody>
<tr>
<td>left, right, straight, left, left, left, straight</td>
<td>GOOD</td>
</tr>
<tr>
<td>left, straight, straight, left, right, straight, straight</td>
<td>BAD</td>
</tr>
<tr>
<td>left, right, straight, left, left, left, straight</td>
<td>18.5</td>
</tr>
<tr>
<td>left, straight, straight, left, right, straight, straight</td>
<td>-3</td>
</tr>
</tbody>
</table>

Given a sequence of examples/states and a reward after completing that sequence, learn to predict the action to take in for an individual example/state

Reinforcement learning example

Backgammon

WIN!

LOSE!

Given sequences of moves and whether or not the player won at the end, learn to make good moves
Reinforcement learning example

http://www.youtube.com/watch?v=VCdxqn0fcnE

Other learning variations

What data is available:
- Supervised, unsupervised, reinforcement learning
- Semi-supervised, active learning, …

How are we getting the data:
- Online vs. offline learning

Type of model:
- Generative vs. discriminative
- Parametric vs. non-parametric

Neural Networks

Neural Networks try to mimic the structure and function of our nervous system

People like biologically motivated approaches

Our Nervous System

What do you know?
Our nervous system: the computer science view

- The human brain is a large collection of interconnected neurons
- A NEURON is a brain cell
  - Collect, process, and disseminate electrical signals
  - Neurons are connected via synapses
  - They fire depending on the conditions of the neighboring neurons

Our nervous system

- The human brain contains $\sim 10^{11}$ (100 billion) neurons
- Each neuron is connected to $\sim 10^4$ (10,000) other neurons
- Neurons can fire as fast as $10^{-3}$ seconds

How does this compare to a computer?

Man vs. Machine

- $10^{11}$ neurons vs. $10^{10}$ transistors
- $10^{11}$ neurons vs. $10^{11}$ bits of ram/memory
- $10^{14}$ synapses vs. $10^{13}$ bits on disk
- $10^{-3}$ “cycle” time vs. $10^{-9}$ cycle time

Brains are still pretty fast

Who is this?
Brains are still pretty fast

If you were me, you’d be able to identify this person in $10^3 \times (1/10)$ s!

Given a neuron firing time of $10^{-3}$ s, how many neurons in sequence could fire in this time?
- A few hundred

What are possible explanations?
- either neurons are performing some very complicated computations
- brain is taking advantage of the massive parallelization

Artificial Neural Networks

Weight $w$ is the strength of signal sent between $A$ and $B$.

If $A$ fires and $w$ is positive, then $A$ stimulates $B$.

If $A$ fires and $w$ is negative, then $A$ inhibits $B$.

A given neuron has many, many connecting, input neurons

If a neuron is stimulated enough, then it also fires.

How much stimulation is required is determined by its threshold.
A Single Neuron/Perceptron

Each input contributes: $x_i \cdot w_i$

Input $x_i$
Weight $w_i$

Input $x_j$
Weight $w_j$

Input $x_k$
Weight $w_k$

Input $x_l$
Weight $w_l$

Threshold function

Output $y$

$in = \sum_i w_i x_i$

Possible threshold functions

**Hard threshold**

$$g(x) = \begin{cases} 1 & \text{if } x > \text{threshold} \\ 0 & \text{otherwise} \end{cases}$$

**Sigmoid**

$$g(x) = \frac{1}{1 + e^{-ax}}$$
A Single Neuron/Perceptron

Threshold of 1

Weighted sum is 0.5, which is not larger than the threshold

1*1 + 0*-1 + 0*1 + 1*0.5 = 1.5

A Single Neuron/Perceptron

Threshold of 1

Weighted sum is 1.5, which is larger than the threshold

Neural network

Individual perceptrons/neurons
Neural network inputs each perceptron computes and calculates an answer.

Those answers become inputs for the next level.

Finally, all levels compute to get the final answer.
Neural networks
Different kinds/characteristics of networks

Feed forward networks

Recurrent network
Output is fed back to input
Can support memory!

How?

Activation spread
http://www.youtube.com/watch?v=Yq7d4ROvZ6I
History of Neural Networks

McCulloch and Pitts (1943) – introduced model of artificial neurons and suggested they could learn
Hebb (1949) – Simple updating rule for learning
Rosenblatt (1962) - the perceptron model
Minsky and Papert (1969) – wrote Perceptrons
Bryson and Ho (1969, but largely ignored until 1980s–Rosenblatt) – invented back-propagation learning for multilayer networks

Perceptron

First wave in neural networks in the 1960’s
Single neuron
Trainable: its threshold and input weights can be modified
If the neuron doesn’t give the desired output, then it has made a mistake.
Input weights and threshold can be changed according to a learning algorithm

Examples - Logical operators

**AND** – if all inputs are 1, return 1, otherwise return 0
**OR** – if at least one input is 1, return 1, otherwise return 0
**NOT** – return the opposite of the input
**XOR** – if exactly one input is 1, then return 1, otherwise return 0

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>$x_1 \text{ and } x_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
### AND

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_1$ and $x_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Input $x_1$, $W_1 = ?$

Input $x_2$, $W_2 = ?$

Input $x_3$, $W_3 = ?$

Input $x_4$, $W_4 = ?$

$T = ?$ 

Output $y$

---

Inputs are either 0 or 1

---

### AND

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_1$ and $x_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Input $x_1$, $W_1 = 1$

Input $x_2$, $W_2 = 1$

Input $x_3$, $W_3 = 1$

Input $x_4$, $W_4 = 1$

$T = 2$ 

Output $y$

Output is 1 only if all inputs are 1

---

Inputs are either 0 or 1

---

### AND

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_1$ and $x_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Input $x_1$, $W_1 = 1$

Input $x_2$, $W_2 = 1$

Input $x_3$, $W_3 = 1$

Input $x_4$, $W_4 = 1$

$T = 4$ 

Output $y$

Output is 1 only if all inputs are 1

---

Inputs are either 0 or 1
**OR**

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_1 \text{ or } x_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Inputs are either 0 or 1

Output is 1 if at least 1 input is 1

Output $y$
Inputs are either 0 or 1

Output is 1 if at least 1 input is 1

OR

NOT

Inputs are either 0 or 1

If input is 1, output is 0.
If input is 0, output is 1.
How about…

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$ and $x_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Training neural networks

Learn individual weights between nodes (e.g. threshold)

Positive or negative?

NEGATIVE

Positive or negative?

NEGATIVE
Positive or negative?

NEGATIVE

Positive or negative?

POSITIVE

A method to the madness

blue = positive

yellow triangles = positive

all others negative

How did you figure this out (or some of it)?

Training neural networks

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$, and $x_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

1. start with some initial weights and thresholds
2. show examples repeatedly to NN
3. update weights/thresholds by comparing NN output to actual output
Perceptron learning algorithm

repeat until you get all examples right:

- for each “training” example:
  - calculate current prediction on example
  - if wrong:
    - update weights and threshold towards getting this example correct

What could we adjust to make it right?

Weighted sum is 0.5, which is not equal or larger than the threshold

This weight doesn’t matter, so don’t change

Could increase any of these weights
Perceptron learning

A few missing details, but not much more than this

Keeps adjusting weights as long as it makes mistakes

If the training data is linearly separable the perceptron learning algorithm is guaranteed to converge to the “correct” solution (where it gets all examples right)

Linearly Separable

A data set is linearly separable if you can separate one example type from the other

Which of these are linearly separable?

Which of these are linearly separable?
Perceptrons

1969 book by Marvin Minsky and Seymour Papert

The problem is that they can only work for classification problems that are linearly separable

Insufficiently expressive

"Important research problem" to investigate multilayer networks although they were pessimistic about their value