Course 16:198:520: Introduction To Artificial Intelligence
Lecture 11

Introduction To Machine Learning

Abdeslam Boularias

Wednesday, March 25, 2020

RUTGERS
THE STATE UNIVERSITY
OF NEW JERSEY

Outline

1. What is Machine Learning?
2. Project
3. Basic Algorithms

What is Intelligence?

Intelligence is a goal-directed adaptive behavior.

What is Intelligence?

Intelligence is a goal-directed adaptive behavior.

An intelligent behavior is:

- Goal-directed: search and inference.
- Adaptive: learning from observations.

What is Intelligence?

Intelligence is a goal-directed adaptive behavior.

An intelligent behavior is:

- Goal-directed: search and inference.
- Adaptive: learning from observations.

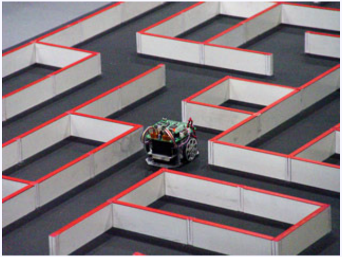
Search and Inference have been covered in the previous lectures

- Search: based on the current knowledge of a problem, what is the best sequence of actions to solve it?
- Inference: based on the current knowledge of a problem, what is the probability of some event?

Where does the knowledge about a problem come from?

Adaptive Behavior

- In the first assignment, you wrote a program for a robot that searches for a goal in a maze.
- The robot can see only nearby obstacles.
- The robot iterates between:
  1. Searching for a path based the current knowledge.
  2. Following the path while simultaneously **learning** about new obstacles and updating the current knowledge.
- This is a goal-directed adaptive behavior.

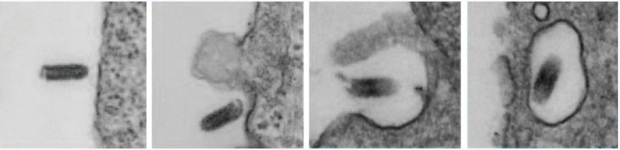


Robot in a maze (©Robo Bionic)

Adaptive Behavior

In order to act successfully in a complex environment, biological systems have developed adaptive behaviors through learning and evolution.

Sunflowers tracking the sun.



The Ebola virus entering a cell.

What is Learning?

Kupfermann (1985)

Learning is the acquisition of knowledge about the world.

What is Learning?

Kupfermann (1985)

Learning is the acquisition of knowledge about the world.

Problem

Is learning simply a process of memorizing knowledge?

What is Learning?

Kupfermann (1985)

Learning is the acquisition of knowledge about the world.

Problem

Is learning simply a process of memorizing knowledge?

Shepherd (1988)

Learning is an adaptive change in behavior caused by experience.

What is Learning?

Kupfermann (1985)

Learning is the acquisition of knowledge about the world.

Problem

Is learning simply a process of memorizing knowledge?

Shepherd (1988)

Learning is an adaptive change in behavior caused by experience.

Notice that the algorithm implemented in Assignment 1 perfectly fits this definition.

What is Machine Learning?

Ron Kohavi; Foster Provost (1998). "Glossary of terms"

Machine Learning is a subfield of computer science that explores the study and construction of **algorithms** that can learn from and make predictions on **data**.

Machine Learning searches for patterns (regularities) in data that allows the prediction of new data.
This is also known as *empirical inference*.

Empirical Inference

Data, observations $\Rightarrow$ rules, models

What is Machine Learning?

Empirical Inference

Data, observations $\Rightarrow$ rules, models

How is machine learning different statistics?

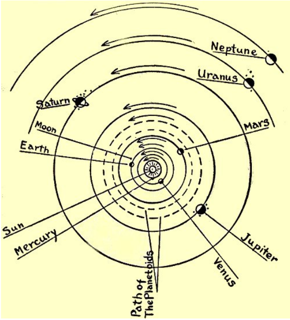
- Both machine learning and statistics are concerned with summarizing data (or extracting rules from data).
- Statistics focus on data analysis (e.g, hypothesis testing).
- Machine learning is more concerned with finding efficient algorithms: algorithms that run fast and require as little data as possible to make predictions that are as accurate as possible.

Empirical Inference

Data, observations $\Rightarrow$ rules, models

Extracting rules from observations has always been the quest of science.

Example



Observations of the movements of planets
(from *The Boy Scientist*)

$\Longrightarrow$

$$F = \frac{GM_1M_2}{r^2}$$

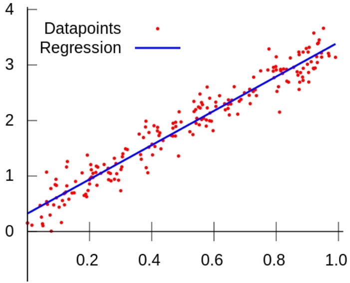
Law of gravity

Empirical Inference

Data, observations $\Rightarrow$ rules, models

Extracting rules from observations has always been the quest of science.

Example



Observations of data points (x, y)

$\Longrightarrow$

$$Y = aX + b$$

Law describing the relation between x and y.

Empirical Inference

Data, observations $\Rightarrow$ rules, models

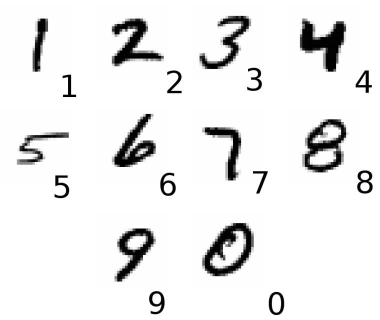
Extracting rules from observations has always been the quest of science.

- Is machine learning the automatization of science?
- Physics searches for laws explaining simple observations about the universe.
- Machine learning searches for laws explaining complex observations, such as protein structures, gene expressions, speech, text, and images.
- For example, machine learning is increasingly becoming an essential tool in biology.

Empirical Inference

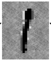
Data, observations $\Rightarrow$ rules, models

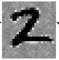
Example: Extract a law that maps an image to a digit



Observations of data points $(image, digit)$

$\Rightarrow$

$f\Big($  $\Big) = 1$

$f\Big($  $\Big) = 2$

$f\Big($  $\Big) = 3$

$\vdots$

Law f describing the relation between $images$ and $digits$.

Empirical Inference

Data, observations $\Rightarrow$ rules, models

Generalization

The rule (or model) should be used to predict new observations.

Generalization

Example

- Observe: $1, 2, 4, 7, \ldots$
- What is next?

Generalization

Example

- Observe: $1, 2, 4, 7, \ldots$
- What is next?
- $1, 2, 4, 7, 11, 16, \ldots$: $a_{n+1} = a_n + n$
- $1, 2, 4, 7, 12, 20, \ldots$: $a_{n+2} = a_{n+1} + a_n + 1$
- $1, 2, 4, 7, 14, 28, \ldots$: divisors of 28.
- $1, 2, 4, 7, 1, 1, 5 \ldots$: decimal expansions of $\pi = 3.14159\ldots$ and $e = 2.718\ldots$ interleaved.
- Which of these answers is the right one?

Generalization

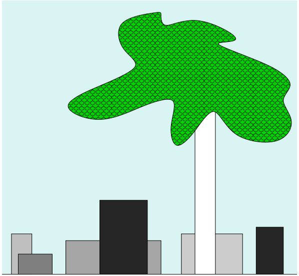
Example

- Observe: $1, 2, 4, 7, \ldots$
- What is next?
- $1, 2, 4, 7, 11, 16, \ldots$: $a_{n+1} = a_n + n$
- $1, 2, 4, 7, 12, 20, \ldots$: $a_{n+2} = a_{n+1} + a_n + 1$
- $1, 2, 4, 7, 14, 28, \ldots$: divisors of 28.
- $1, 2, 4, 7, 1, 1, 5 \ldots$: decimal expansions of $\pi = 3.14159\ldots$ and $e = 2.718\ldots$ interleaved.
- Which of these answers is the right one?
- We don't know.

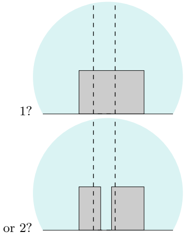
Principle of Occam's razor

Among competing hypotheses, the one with the fewest assumptions should be selected.
In other terms, the simplest model (or rule) is the one that will most likely make the smallest generalization errors.

Example

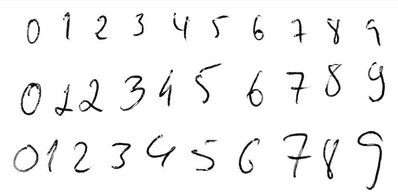


Observation

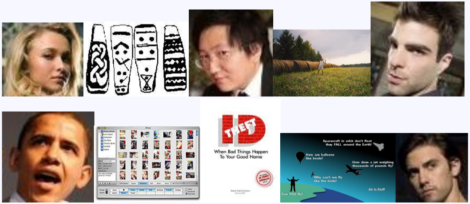


Which hypothesis is true, 1 or 2?

from David J.C. MacKay. *Information Theory, Inference, and Learning Algorithms*.



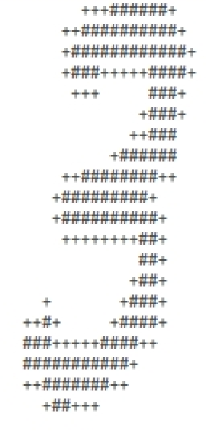
Which Digit?



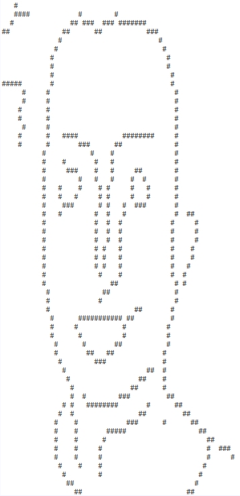
Which are Faces?

Project: Image Classification

Data: `http://rl.cs.rutgers.edu/fall2019/data.zip`



Which Digit?



Face or not face?

Project: Image Classification

What you should do

1. Implement three classification algorithms for detecting faces and classifying digits:
   - Perceptron
   - Naive Bayes Classifier
   - Optional: an algorithm of your choice (k-nearest neighbors, support vector machine, random forests, neural networks)
2. Design the features for each of the two problems.
3. Compare the three algorithms, and report the prediction error (and standard deviation) as a function of the number of data points used for trainning.
4. Write a small report (minimum two pages) describing the implemented algorithms and discussing the results.
5. Submit the code and the report by December 9^{th}, 2019.
6. Setup an appointement with the TA for demonstrating your submitted program

Project: Image Classification

- Do NOT use an existing library for the learning algorithms!
- It's OK to share ideas, but not code or writing.
- Part of your score will depend on the accuracy of the predictions made by your program.
- The data set is separated into three sets:
  - Training and validation: used to learn and find the parameters of your model.
  - Testing: used to evaluate the learned model.
- Your algorithm should not look at the testing data before the training is over. If you use any testing data point for training, that would be considered as cheating.

Project: Image Classification

Acknowledgement: This topic is based on the one created by Dan Klein and John DeNero that was given as part of the programming assignments of Berkeley's CS188 course.

https://www.cs.utexas.edu/~pstone/Courses/343spring10/assignments/classification/classification.html

http://inst.eecs.berkeley.edu/~cs188/sp11/projects/classification/classification.html

- The set of training examples is defined as $\mathcal{X}_{train} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$. Where x_i is an input (image, text, etc.) and y_i is a label (e.g, face or not face).
- The training set is provided to the algorithm.
- The algorithm learns a function f such that $f(x_i) = y_i$ for most of the training examples.
- The performance of the algorithm is evaluated according to the accuracy of its predictions on a testing set $\mathcal{X}_{test} = \{(x_{n+1}, y_{n+1}), (x_{n+2}, y_{n+2}), \ldots, (x_m, y_m)\}$ that is not known during the training.

Naive Bayes Algorithm

Recall Bayes Rule. We could use it to infer:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

We may dispose of the requirement of knowing $p(x)$ by settling for a likelihood ratio

$$L(x) = \frac{p(y = true|x)}{p(y = false|x)} = \frac{p(x|y = true)p(y = true)}{p(x|y = false)p(y = false)},$$

and deciding $y = true$ if $L(x) \geq 1$ and $y = false$ if $L(x) < 1$

So, all we need is to estimate $p(x|y)$ and $p(y)$ from the training examples that we have.

Naive Bayes Algorithm

Estimating $p(y)$ from the training examples
$\mathcal{X}_{train} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$

$$p(y = true) = \frac{\text{Number of times } y_i = true \text{ in } \mathcal{X}_{train}}{\text{n}}$$
$$p(y = false) = \frac{\text{Number of times } y_i = false \text{ in } \mathcal{X}_{train}}{\text{n}}$$

We will see now how to estimate $p(x|y)$ from the training examples $\mathcal{X}_{train} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$.

First define a function ϕ that maps each input x into a features vector $\phi(x) = \big(\phi_1(x), \phi_2(x), \ldots, \phi_l(x)\big)$.
Example 1 (very raw):
$\phi(x) = \big(\text{pixel 1 of } x, \text{pixel 2 of } x, \ldots, \text{pixel l of } x\big)$.
Example 2 (very basic):
$\phi(x) = \big(\text{number of black pixels of } x, \text{number of white pixels of } x\big)$.

Then, we estimate the probability distribution of each feature ϕ_j:

$$p(\phi_j(x) = v | y = true) = \frac{\text{Nb. of times } \phi_j(x) = v \text{ and } y = true \text{ in } \mathcal{X}_{train}}{\text{Nb. of times } y = true \text{ in } \mathcal{X}_{train}}$$

$$p(\phi_j(x) = v | y = false) = \frac{\text{Nb. of times } \phi_j(x) = v \text{ and } y = false \text{ in } \mathcal{X}_{train}}{\text{Nb. of times } y = false \text{ in } \mathcal{X}_{train}}$$

Naive Bayes Algorithm

Example: $\phi_1(x)$ is the number of black pixels of an image x, and y indicates if x is a face or not.

$$p(\phi_1(x) = 0 | y = true) = \frac{\text{Number of face images with 0 black pixels}}{\text{Number of face images}} = 0.09$$

$$p(\phi_1(x) = 1 | y = true) = \frac{\text{Number of face images with 1 black pixel}}{\text{Number of face images}} = 0.13$$

$$p(\phi_1(x) = 2 | y = true) = \frac{\text{Number of face images with 2 black pixels}}{\text{Number of face images}} = 0.24$$

$$p(\phi_1(x) = 3 | y = true) = \frac{\text{Number of face images with 3 black pixels}}{\text{Number of face images}} = 0.18$$

$$...etc.$$

Naive Bayes Algorithm

$\phi_2(x)$ is the number of white pixels of an image x, and y indicates if x is a face or not.

$$p(\phi_2(x) = 0 | y = true) = \frac{\text{Number of face images with 0 white pixels}}{\text{Number of face images}} = 0.10$$
$$p(\phi_2(x) = 1 | y = true) = \frac{\text{Number of face images with 1 white pixel}}{\text{Number of face images}} = 0.31$$
$$p(\phi_2(x) = 2 | y = true) = \frac{\text{Number of face images with 2 white pixels}}{\text{Number of face images}} = 0.02$$
$$p(\phi_2(x) = 3 | y = true) = \frac{\text{Number of face images with 3 white pixels}}{\text{Number of face images}} = 0.11$$
$$...etc.$$

Naive Bayes Algorithm

Finally, we estimate $p(x|y)$ using the *naive Bayes assumption*

$$p(x|y = true) = \prod_{j=1}^{l} p(\phi_j(x)|y = true)$$

$$p(x|y = false) = \prod_{j=1}^{l} p(\phi_j(x)|y = false)$$

This is called naive Bayes because we assume, given the label, the features are independent of each other.

Example

In the previous example, assume that $x \in \mathcal{X}_{test}$ is a test image with 2 black pixel and 3 white pixels, then $p(x|y = true) = p(\phi_1(x) = 2|y = true)p(\phi_2(x) = 3|y = true) = 0.24 \times 0.11$

- The perceptron algorithm was invented in 1957 at the Cornell Aeronautical Laboratory by Frank Rosenblatt funded by the United States Office of Naval Research.
- The Perceptron is a single-layer neural network, modern deep neural nets are nothing but Perceptrons stacked on top of each other.
- The idea is to learn a linear decision function f defined as: $f(x_i, w) = w_0 + w_1\phi_1(x_i) + w_2\phi_2(x_i) + w_3\phi_3(x_i) + \cdots + w_l\phi_l(x_i)$, and given a new test point x, predict its label $y = true$ if $f(x_i, w) \geq 0$ and $y = false$ if $f(x_i, w) < 0$.

Perceptron Algorithm

Steps

1. Initialize the weights $\{w_j\}$. Weights may be initialized to 0 or to a small random value, this does not matter.
2. For each example (x_i, y_i) in our training set $\mathcal{X}_{train}$, do:
   - Compute $f(x_i, w) = w_0 + w_1\phi_1(x_i) + w_2\phi_2(x_i) + w_3\phi_3(x_i) + \cdots + w_l\phi_l(x_i)$
   - If $f(x_i, w) \geq 0$ and $y_i = true$ or $f(x_i, w) < 0$ and $y_i = false$, then do nothing, just move to the next example (x_{i+1}, y_{i+1})

Perceptron Algorithm

Steps

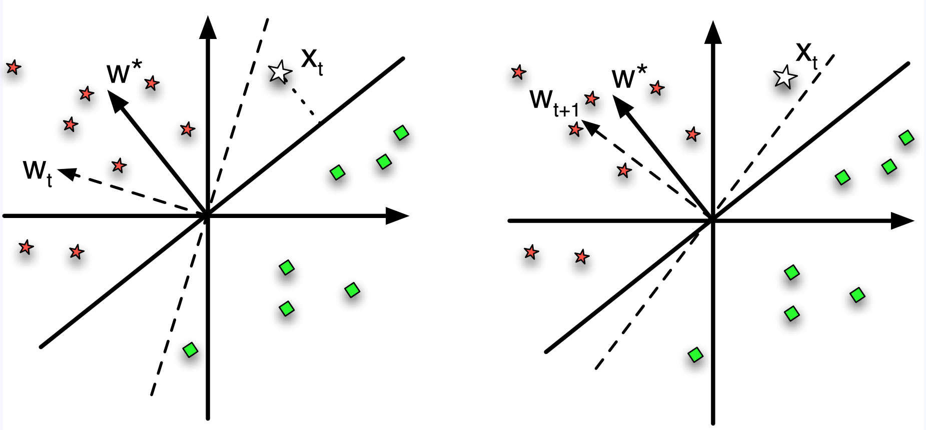
1. Initialize the weights $\{w_j\}$. Weights may be initialized to 0 or to a small random value, this does not matter.
2. For each example (x_i, y_i) in our training set $\mathcal{X}_{train}$, do:
    - Compute $f(x_i, w) = w_0 + w_1\phi_1(x_i) + w_2\phi_2(x_i) + w_3\phi_3(x_i) + \cdots + w_l\phi_l(x_i)$
    - If $f(x_i, w) \geq 0$ and $y_i = true$ or $f(x_i, w) < 0$ and $y_i = false$, then do nothing, just move to the next example (x_{i+1}, y_{i+1})
    - Else, update the weights $\{w_j\}$:
        - If $f(x_i, w) < 0$ and $y_i = true$ then: $w_j \leftarrow w_j + \phi_j(x_i)$, for $j = 1, \ldots, l$, and $w_0 \leftarrow w_0 + 1$
        - If $f(x_i, w) \geq 0$ and $y_i = false$ then: $w_j \leftarrow w_j - \phi_j(x_i)$, for $j = 1, \ldots, l$, and $w_0 \leftarrow w_0 - 1$

Perceptron Algorithm

Steps

1. Initialize the weights $\{w_j\}$. Weights may be initialized to 0 or to a small random value, this does not matter.
2. For each example (x_i, y_i) in our training set $\mathcal{X}_{train}$, do:
   - Compute $f(x_i, w) = w_0 + w_1\phi_1(x_i) + w_2\phi_2(x_i) + w_3\phi_3(x_i) + \cdots + w_l\phi_l(x_i)$
   - If $f(x_i, w) \geq 0$ and $y_i = true$ or $f(x_i, w) < 0$ and $y_i = false$, then do nothing, just move to the next example (x_{i+1}, y_{i+1})
   - Else, update the weights $\{w_j\}$:
     - If $f(x_i, w) < 0$ and $y_i = true$ then: $w_j \leftarrow w_j + \phi_j(x_i)$, for $j = 1, \ldots, l$, and $w_0 \leftarrow w_0 + 1$
     - If $f(x_i, w) \geq 0$ and $y_i = false$ then: $w_j \leftarrow w_j - \phi_j(x_i)$, for $j = 1, \ldots, l$, and $w_0 \leftarrow w_0 - 1$
3. Stop if you made a pass on all the examples $\mathcal{X}_{train}$ without making any updates, or after a certain time limit that you pre-defined. Otherwise, go back to step 2 and repeat.

Perceptron Algorithm



A perceptron updating its linear decision boundary (dashed line) as a new training example x_t is added. W^* is the optimal weights (boundary).