Open-Source Machine Learning for an Embedded System

Decision Trees with Processing and Arduino

Lucas Spicer B.S.EE

spicerrobots.com

http://karenswhimsy.com/tree-clipart.shtm

Machine Learning (ML)

A branch of Artificial Intelligence which deals with algorithms which allow computers to generalize example data probability distributions in order to improve their behaviors

ML traditionally attempts to improve complex relationship recognition from limited data and to provide human intelligible insight into those relationships

Examples: Netflix Suggestions, Google Instant Search, Credit Card Fraud Detection, etc.

The Challenge

Because Machine Learning is traditionally performed on expensive proprietary software systems (like Matlab) the goal for this project is:

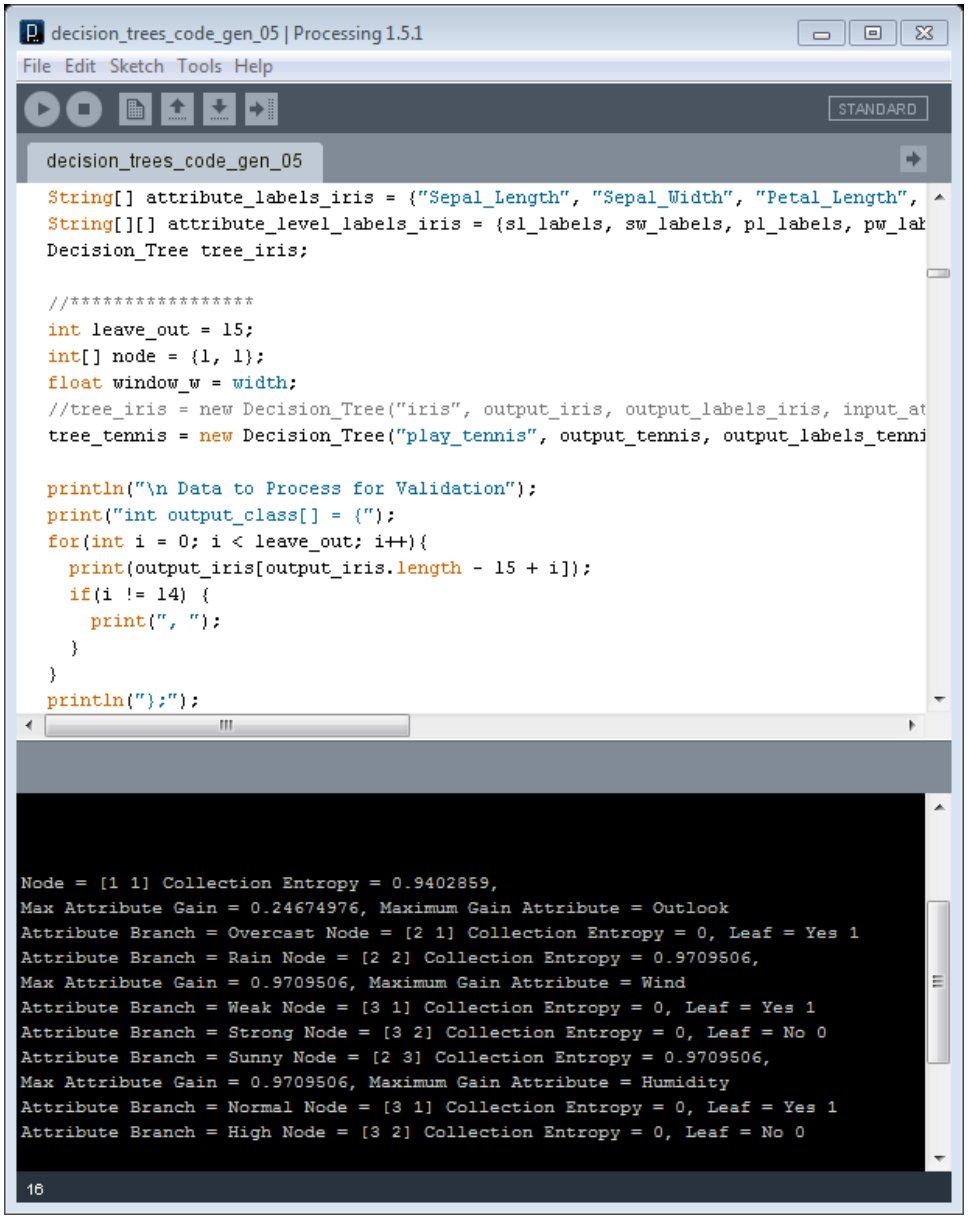
To use open-source free software to generate decision trees from arbitrary numbers of examples with arbitrary numbers of attributes and arbitrary numbers of levels of those attributes, as well as arbitrary numbers of output classes

To provide source-code output to implement the generated decision trees on an open-source low cost embedded development system, as well as human readable or graphical output to explain and educate how the generation process works

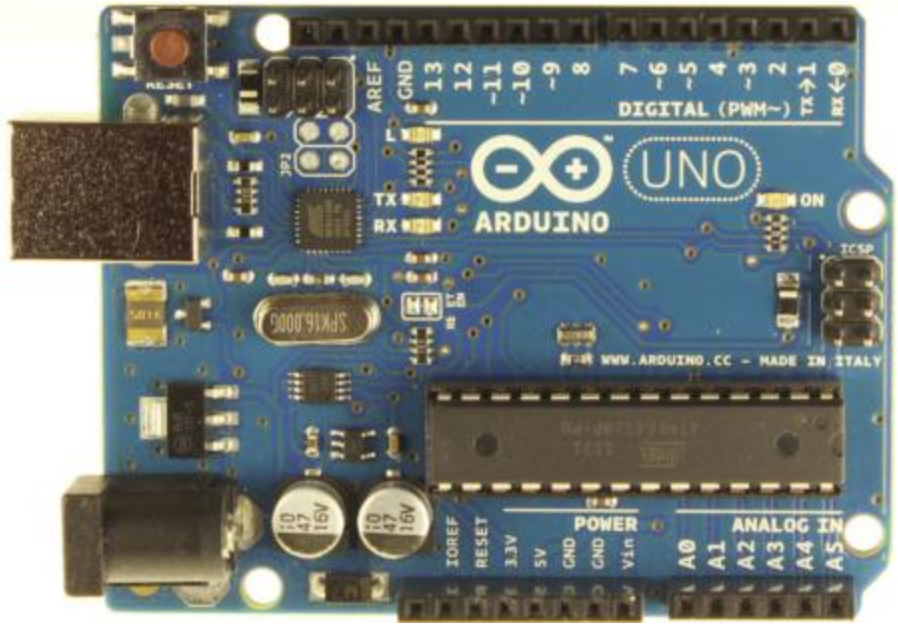
Processing is a free open-source programming language, development environment, and online community that promotes software literacy within the visual arts

Processing was initially created to serve as a software sketchbook and to teach fundamentals of computer programming within a visual context

http://processing.org



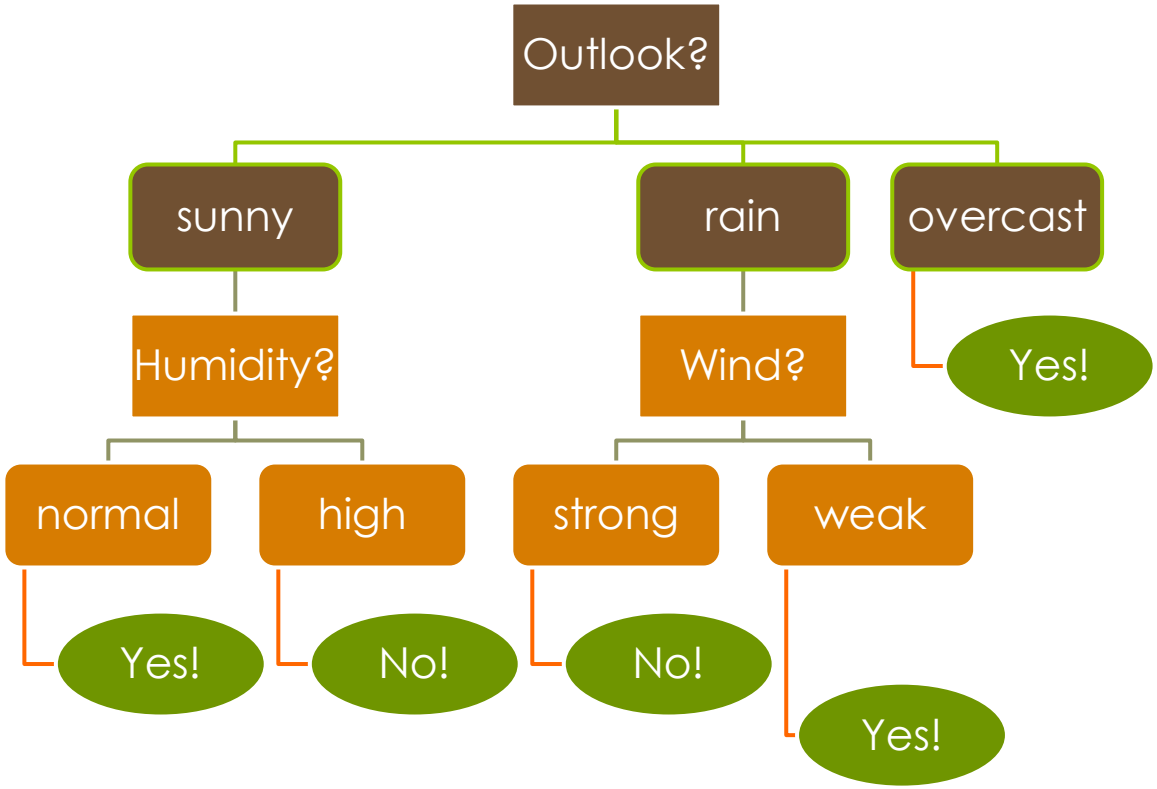
Processing Sketchbook IDE running the Decision Tree Generator



Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments.

http://arduino.cc

Decision Trees



- Root Node
- Nodes (Tests)
- Leaf Nodes (Decisions)

```
function ID3
Input:   (R: a set of non-target attributes,
          C: the target attribute,
          S: a training set) returns a decision tree;
begin    If S is empty, return a single node with
      value Failure;
   If S consists of records all with the same
      value for the target attribute,
      return a single leaf node with that value;
   If R is empty, then return a single node
      with the value of the most frequent of the
      values of the target attribute that are
      found in records of S; [in that case
      there may be be errors, examples
      that will be improperly classified];
   Let A be the attribute with largest
      Gain(A,S) among attributes in R;
   Let {aj| j=1,2, .., m} be the values of
      attribute A;
   Let {Sj| j=1,2, .., m} be the subsets of
      S consisting respectively of records
      with value aj for A;
   Return a tree with root labeled A and arcs
      labeled a1, a2, .., am going respectively
      to the trees (ID3(R-{A}, C, S1), ID3(R-{A}, C, S2),
      .....,ID3(R-{A}, C, Sm);
   Recursively apply ID3 to subsets {Sj| j=1,2, .., m}
      until they are empty
end
```

J. Ross Quinlan's classic Decision Tree Algorithm ID3

Assumes Discrete Data Classes

Recursive Splitting is based on Entropy and Information Gain

Entropy is a Measure of Uncertainty in Data

S is a data set

p_i is the proportion of the set from the i^{th} class of S

Zero Entropy occurs when the entire set is from one class

$$Entropy(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

The concept was introduced by Claude E. Shannon in his 1948 paper "A Mathematical Theory of Communication"

Information Gain is a Reduction in Entropy

S is a data set

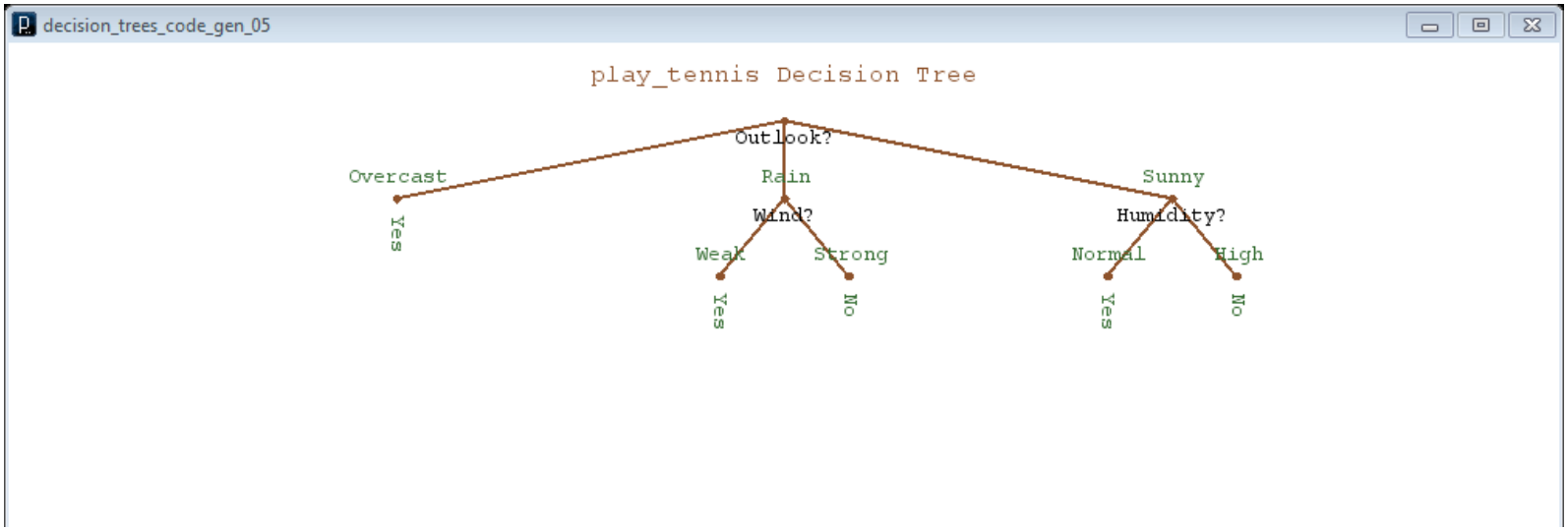
A is a subset of S with a given attribute

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Goal is to test attributes which provide the maximum information gain for a given data set

Example Data Set

Day	Outlook	Temperature	Humidity	Wind	PlayTennis?
1	sunny	hot	high	weak	No
2	sunny	hot	high	strong	No
3	overcast	hot	high	weak	Yes
4	rain	mild	high	weak	Yes
5	rain	cool	normal	weak	Yes
6	rain	cool	normal	strong	No
7	overcast	cool	normal	strong	Yes
8	sunny	mild	high	weak	No
9	sunny	cool	normal	weak	Yes
10	rain	mild	normal	weak	Yes
11	sunny	mild	normal	strong	Yes
12	overcast	mild	high	strong	Yes
13	overcast	hot	normal	weak	Yes
14	rain	mild	high	strong	No
15	sunny	hot	normal	strong	No
16	sunny	hot	normal	strong	Yes




```
Node = [1 1] Collection Entropy = 0.9402859,
Max Attribute Gain = 0.24674976, Maximum Gain Attribute = Outlook
Attribute Branch = Overcast Node = [2 1] Collection Entropy = 0, Leaf = Yes 1
Attribute Branch = Rain Node = [2 2] Collection Entropy = 0.9709506,
Max Attribute Gain = 0.9709506, Maximum Gain Attribute = Wind
Attribute Branch = Weak Node = [3 1] Collection Entropy = 0, Leaf = Yes 1
Attribute Branch = Strong Node = [3 2] Collection Entropy = 0, Leaf = No 0
Attribute Branch = Sunny Node = [2 3] Collection Entropy = 0.9709506,
Max Attribute Gain = 0.9709506, Maximum Gain Attribute = Humidity
Attribute Branch = Normal Node = [3 1] Collection Entropy = 0, Leaf = Yes 1
Attribute Branch = High Node = [3 2] Collection Entropy = 0, Leaf = No 0
```

Example Decision Tree Output

Calculations and Graphical Representation of Tree

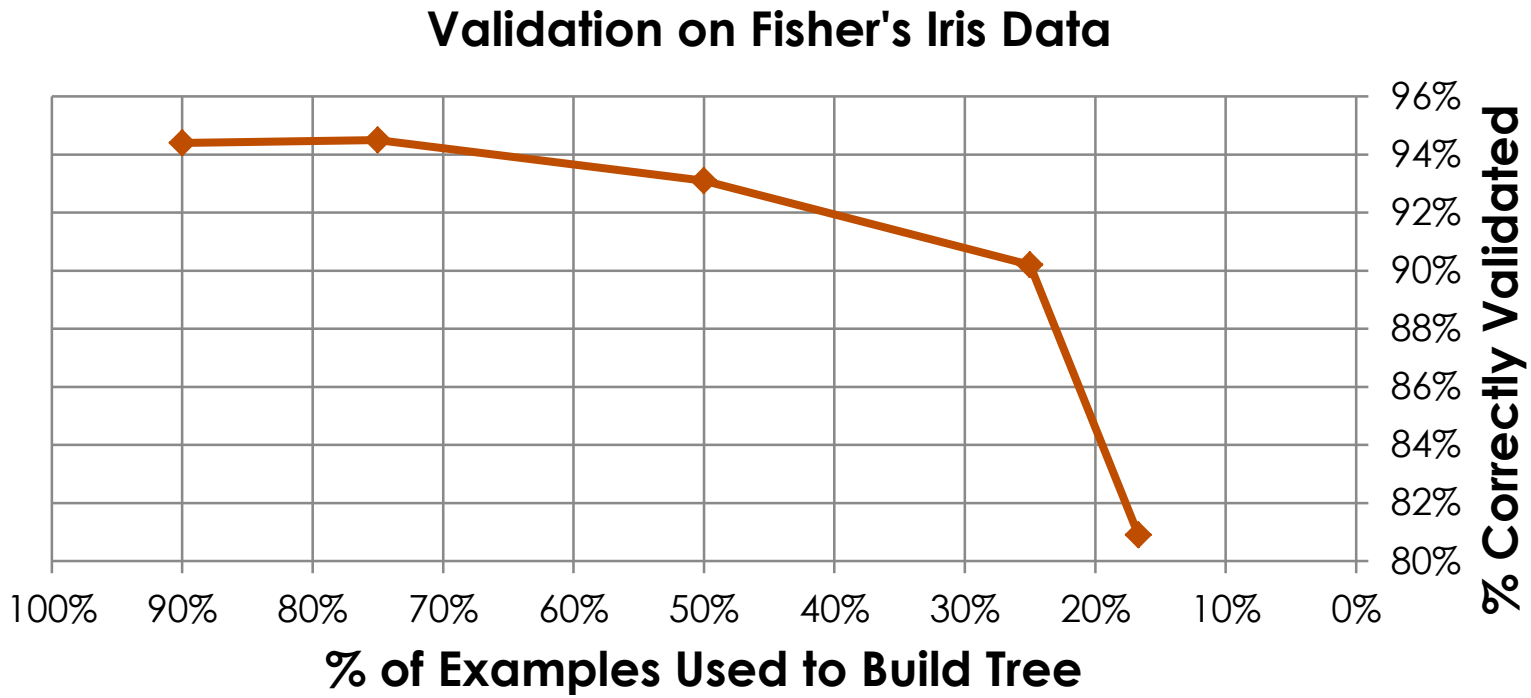
Example Auto-Generated Arduino Function output from Processing

Allows Arduino to implement the tree "grown" (trained) on a computer running Processing

```
/* play_tennis decision Tree file for Arduino   */
/* Date: 28 Feb 2012                            */
/* Time: 22:0:46                                */

#ifndef play_tennis_H
#define play_tennis_H

#if defined(ARDUINO) && ARDUINO >= 100
#include "Arduino.h"
#else
#include "WProgram.h"
#endif

//  Output (Decision):
// play_tennis  levels: 0 = No 1 = Yes
//  Input (Attributes):
// Outlook  levels: 0 = Overcast 1 = Rain 2 = Sunny
// Temp levels: 0 = Cool 1 = Mild 2 = Hot
// Humidity levels: 0 = Normal 1 = High
// Wind levels: 0 = Weak 1 = Strong


int play_tennis(int Outlook, int Temp, int Humidity, int Wind){
    if(Outlook == 0){   // Outlook == Overcast?
        return 1;   // play_tennis = Yes
    }
    else if(Outlook == 1){  // Outlook == Rain?
        if(Wind == 0){  // Wind == Weak?
            return 1;   // play_tennis = Yes
        }
        else if(Wind == 1){ // Wind == Strong?
            return 0;   // play_tennis = No
        }
    }
    else if(Outlook == 2){  // Outlook == Sunny?
        if(Humidity == 0){  // Humidity == Normal?
            return 1;   // play_tennis = Yes
        }
        else if(Humidity == 1){ // Humidity == High?
            return 0;   // play_tennis = No
        }
    }
}

#endif
```

Validation

- Key task for ML systems is to validate their ability to generalize from examples
- Data Set is partitioned into a training set and a validation set.
- Training set is used to build the decision tree and validation set is used to test its ability to generalize



Applications

- Freely and easily available educational tool to instruct about machine learning
- Software library to give small robot hobbyists ability to make smarter, learning robots (or other embedded devices)
- Examples: smart watering can, obstacle avoiding robots, automatic failure diagnosis for small embedded devices, etc.