TRƯỜNG ĐH CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc Lập – Tự do – Hạnh phúc

ĐỒ ÁN: XÂY DỰNG CÂY ĐỊNH DANH

Lớp : CS110.H11.KHTN

Môn : Nhập môn công nghệ tri thức & máy học

GVLT: Ts. Đỗ Văn Nhơn

GVTH: Ths. Nguyễn Đình Hiển

SVTH: 1) Trịnh Mẫn Hoàng – 14520320

2) Phan Đình Nguyên – 14520608

3) Triệu Tráng Vinh – 14521097

TP.Hồ Chí Minh, tháng 01, năm 2017

Contents

1. Giới thiệu bài toán

- Trí tuệ nhân tạo ngày nay đã trở thành một trong những lĩnh vực mũi nhọn, tiên phong hàng đầu, con người chúng ta làm cho máy tính trở nên thông minh hơn, tìm cách tạo ra các chương trình thông minh hơn để có khả năng giải quyết các vấn đề thực tế như các giải quyết của con người.

- Từ khi máy học ra đời, máy tính còn có nghĩa là việc mô hình hóa môi trường xung quanh hay khả năng một chương trình máy tính sinh ra một cấu trúc dữ liệu mới khác với cấu trúc hiện có. Do đó, ra đời mô hình cây định danh dùng để giải quyết các yêu cầu trên.

- Cây định danh là cây mà nếu ta đi từ nút gốc đến các lá thì ta sẽ có một quyết định hay một quy luật dựa vào các thuộc tính trên đường đi từ gốc đến lá, như vậy mỗi đường đi từ nút gốc đến nút lá sẽ cho một quy luật. Vì vậy người ta còn gọi cây định danh là cây quyết định. Ứng dụng nhiều trong lĩnh vực data mining.

- Dưới đây ta sẽ trình bày phương pháp tiếp cận, xây dựng mô hình cây định danh bằng thuật toán đo độ hỗn loạn.

2. Thuật toán độ đo hỗn loạn (entropy)

- Giả sử ta có bộ dữ liệu dataset như sau:

- Nhiệm vụ bài toán là làm sao phân loại từ bảng dữ liệu trên, sang mô hình cây để dễ dàng tiếp cận hình dung quy luật của dữ liệu.

- Thuật toán xây dựng cây quyết định dựa trên giải thuật ID3 bởi J.R. Quinlan, bằng cách tiếp cận theo hướng Top-Down. ID3 sủ dụng tính toán entropy để xây dựng cây định danh

- Ta sẽ tính entropy cho một thuộc tính dựa trên công thức sau:

- Ta sẽ lấy thuộc tính mục tiêu để tính toán trước là Play Golf.

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

Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)
= Entropy (0.36, 0.64)
= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)
= 0.94

- Dựa trên bảng số liệu, ta sẽ đếm tần số xuất hiện của yes là 9 và no là 5. Khi đó tỉ lệ yes so với tổng thể là 9/14 = 0.64, và no là 5/14 = 0.36. Lần lượt thế các số trên vào theo công thức ta được kết quả là 0.94.
- Tiếp đến sẽ tính entropy dựa trên 2 thuộc tính:

$$E(T, X) = \sum_{c \in X} P(c) E(c)$$

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

E(PlayGolf, Outlook) = **P**(Sunny)***E**(3,2) + **P**(Overcast)***E**(4,0) + **P**(Rainy)***E**(2,3)
= (5/14)*0.971 + (4/14)*0.0 + (5/14)*0.971
= 0.693

- Giả sử ta có Outlook gồm 3 giá trị: Sunny, Overcast, Rainy. Giả sử ta xét Outlook là Sunny, ta sẽ lọc ra những phần tử cần có Overcast là Sunny, và thống kê lại như bản trên. Với Sunny, ta có 3 yes, 2 no. Ta sẽ tính entropy (3, 2) cho 1 thuộc tính như trên hàm E(S). Sau đó lại tính tỉ lệ của Sunny so với tổng thể tất là sunny sẽ có 5 phần tử, và p = 5/14. Lần lượt tính như vậy đối với các giá trị Overcast, Rainy.
- Dưới đây là thuật toán xây dựng:
    - **Bước 1**: Tính lượng entropy cho thuộc tính đích.

Entropy(PlayGolf) = Entropy (5,9)
= Entropy (0.36, 0.64)
= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)
= 0.94

- **Bước 2:** Ta sẽ chọn thuộc tính nào cần làm nút cho cây. Bằng cách tính toán số liệu dựa trên hàm Gain(T,X) như sau.

$$Gain(T,X) = Entropy(T) - Entropy(T,X)$$

G(PlayGolf, Outlook) = **E**(PlayGolf) – **E**(PlayGolf, Outlook)

= 0.940 – 0.693 = 0.247

- Ta lần lượt tính cho cách thuộc tính khác, và được bảng số liệu sau:

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

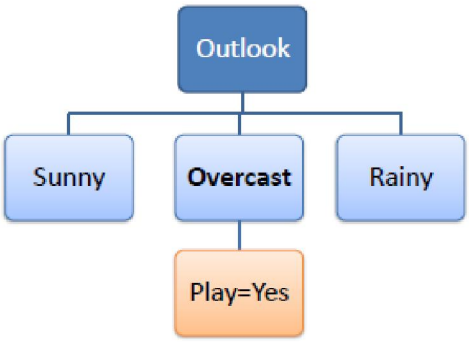
		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			

- **Bước 3:** Chọn thuộc tính có giá trị Gain lớn nhất. Ở đây sẽ là Outlook.

★		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

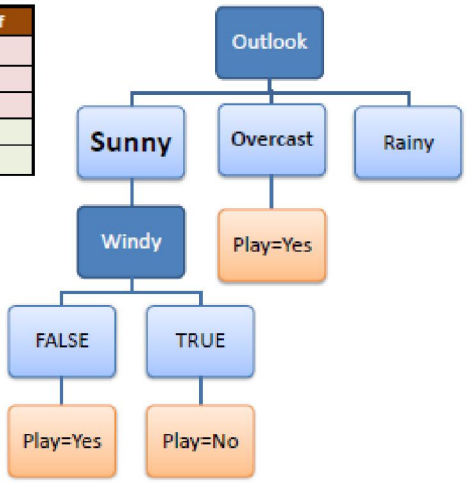
- **Bước 4:** Ta có giá trị entropy của Overcast là 0 (trong phần tính toán hàm E(T,X)). Nên xét Overcast là nút lá, và suy ra được kết quả là Yes.

Temp	Humidity	Windy	Play Golf
Hot	High	FALSE	Yes
Cool	Normal	TRUE	Yes
Mild	High	TRUE	Yes
Hot	Normal	FALSE	Yes
Hot	High	FALSE	Yes



- **Bước 5:** Đối với nhưng giá trị entropy cảu Sunny, Rainy khác 0, nên ta sẽ tiếp tục phân loại cho mỗi thành phần này. Bằng cách lọc ra các phần tử là Sunny, Rainy. Như dưới đây ta sẽ lọc ra cho Sunny.

Temp	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No



- **Bước 6:** Lại tiếp tục quay về bước 1, đệ quy đến khi phân loại hết dữ liệu.

- Cuối cùng ta có bảng hoàn chỉnh như sau.

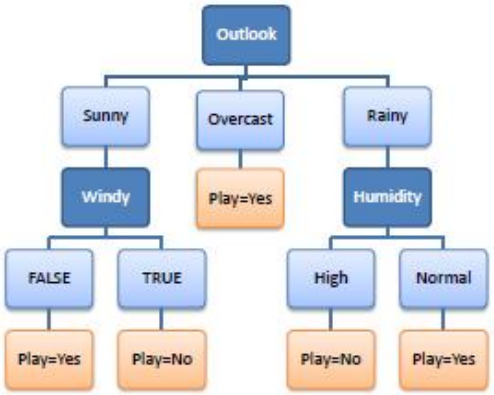
R_1: IF (Outlook=Sunny) AND (Windy=FALSE) THEN Play=Yes

R_2: IF (Outlook=Sunny) AND (Windy=TRUE) THEN Play=No

R_3: IF (Outlook=Overcast) THEN Play=Yes

R_4: IF (Outlook=Rainy) AND (Humidity=High) THEN Play=No

R_5: IF (Outlook=Rain) AND (Humidity=Normal) THEN Play=Yes



3. Thuật giải chi tiết và cài đặt.

- Ứng dụng được xây dựng trên nền tảng Web, sử dụng ngôn ngữ chính là HTMl + CSS Javascript, và một số thư viện giao diện như SemanticUI, GoJS, AngularJS …

- Để xây dựng cây định danh ta sẽ sử dụng cấu trúc dữ liệu JSON (cấu trúc từ điển), cấu trúc này rất thuận tiện trong quá trình lưu mảng, thuộc tính, giá trị…

- Việc lưu trữ trên dữ liệu cũng khá đơn giản và dễ sử dụng.

- Dưới đây là dataset mẫu dưới dạng cấu trúc JSON.

```
test3: {
  rawList: [
    ["Male", "0", "Cheap", "Low", "Bus"],
    ["Male", "1", "Cheap", "Medium", "Bus"],
    ["Female", "1", "Cheap", "Medium", "Train"],
    ["Female", "0", "Cheap", "Low", "Bus"],
    ["Male", "1", "Cheap", "Medium", "Bus"],
    ["Male", "0", "Standard", "Medium", "Train"],
    ["Female", "1", "Standard", "Medium", "Train"],
    ["Female", "1", "Expensive", "High", "Car"],
    ["Male", "2", "Expensive", "Medium", "Car"],
    ["Female", "2", "Expensive", "High", "Car"],
  ],
  field: ["Gender", "CarOwner", "TravelCost", "Income", "Transportation"],
  listAttribute: [
    {
      Name: "Gender",
      Values: ["Male", "Female"]
    },
    {
      Name: "CarOwner",
      Values: ["0", "1", "2"]
    },
    {
      Name: "TravelCost",
      Values: ["Cheap", "Standard", "Expensive"]
    },
    {
      Name: "Income",
      Values: ["Low", "Medium", "High"]
    },
  ],
  goalAttr: {
    Name: "Transportation",
    Values: ["Bus", "Train", "Car"]
  },
},
```

- Đầu tiên ta sẽ xây dựng hàm tìm entropy cho một thuộc tính.

```
var entropyOneAttr = function(list, attr) {
  var sum = list.length;
  var result = 0;

  attr.Values.forEach(function(value) {
    var freq = list.reduce(function(sum, item) {
      if (item[attr.Name] == value)
        return sum + 1;
      else
        return sum;
    }, 0);
    var p = freq / sum;
    var temp = p

    if (p != 0) {  // the attribute does not exist in list
      result = result - p * Math.log2(p);
    }

  })

  return result;
}
```

- Dữ liệu đầu vào sẽ là list (data), và attr (chứa những giá trị của thuộc tính). Ta lần lượt tính cho từng giá trị trong attr theo công thức E(S) ở trên để ra giá trị entropy.

- Tiếp đến ta lại xây dựng hàm tính entropy cho 2 thuộc tính như hàm E(T,X) ở trên.

```
var entropyTwoAttr = function(list, attrGoal, attrElem) {
  var sum = list.length;
  var listEntropy = [];
  var result = 0;
  attrElem.Values.forEach(function(value) {
    var filterList = list.filter(function(item) {
      return item[attrElem.Name] == value
    })
    var lengthFilterList = filterList.length;
    var pValue = lengthFilterList / sum;
    var entropy = entropyOneAttr(filterList, attrGoal);

    if (! isNaN(entropy))
      result = result + pValue * entropy;

    listEntropy.push({
      value: value,
      entropy: entropy
    })

  })

  return {
    value: result,
    listEntropy: listEntropy
  }

}
```

- Sau khi xây dựng xong 2 hàm trên. Ta sẽ bắt phân loại và chọn thuộc tính nào làm node dựa trên giá trị Gain, handleClassifyTree sẽ xử lý công việc này. Ngoài ra còn hàm classifyRecursive dùng để thực hiện lời gọi đệ quy và kiểm tra tất cả các phần tử đã được phân loại hoàn tất. Vì code khá dài nên không tiện viết ở đây. Có thể xem thêm tại "decisionTree/deploy/decision-tree.js".

- Sau quá trình trên ta có được cấu trục dữ liệu được phân loại như sau:

```
{
  "Name": "Outlook",
  "listEntropy": [
    {
      "value": "Sunny",
      "entropy": 0.9709505944546686,
      "child": {
        "Name": "Windy",
        "listEntropy": [
          {
            "value": "True",
            "entropy": 0,
            "finalResult": {
              "Name": "PlayGolf",
              "Value": "No"
            }
          },
          {
            "value": "False",
            "entropy": 0,
            "finalResult": {
              "Name": "PlayGolf",
              "Value": "Yes"
            }
          }
        ]
      }
    },
```

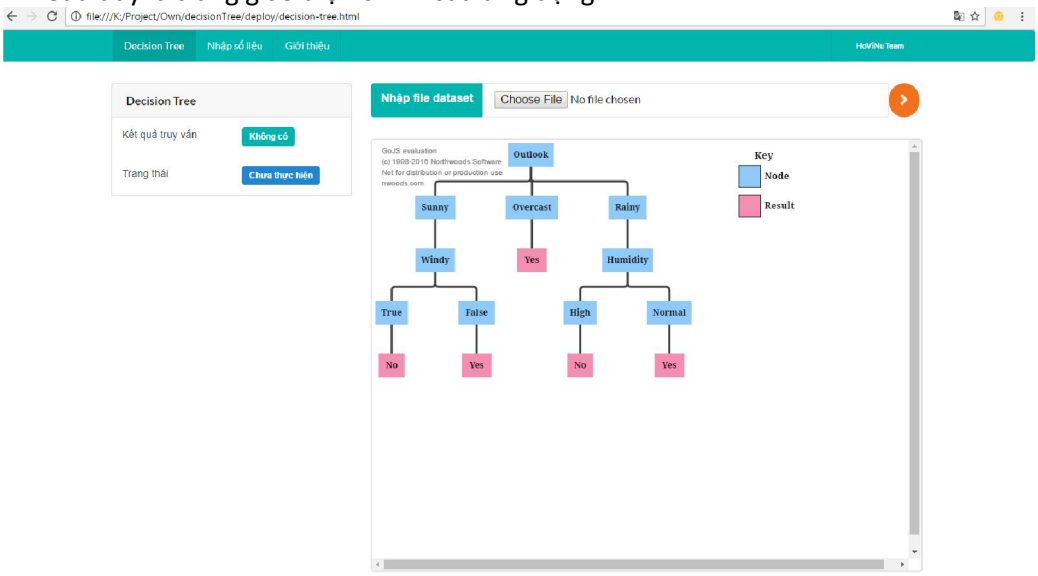
```
      {
        "value": "Overcast",
        "entropy": 0,
        "finalResult": {
          "Name": "PlayGolf",
          "Value": "Yes"
        }
      },
      {
        "value": "Rainy",
        "entropy": 0.9709505944546686,
        "child": {
          "Name": "Humidity",
          "listEntropy": [
            {
              "value": "High",
              "entropy": 0,
              "finalResult": {
                "Name": "PlayGolf",
                "Value": "No"
              }
            },
            {
              "value": "Normal",
              "entropy": 0,
              "finalResult": {
                "Name": "PlayGolf",
                "Value": "Yes"
              }
            }
          ]
        }
      }
    ]
}
```


- Giả sử ta có bộ test cần biết kết quả thông qua cây định danh, hàm predictResultByTree sẽ xử lý việc này. VIệc xử lý khá đơn giản với cấu trúc dữ liệu JSON, ta không cần phải xây dựng tập luật. Ta chỉ việc tìm từng thuộc tính kết quả so khớp với test, nếu có thuộc trường đó, thì lại tiếp tục đệ quy vào cấu trúc con là 'child', rồi lại tiếp tục tìm đến khi kết thúc và trả về kết quả.

```
var predictResultByTree = function(classifyTree, object) {

  var rootAttr = classifyTree.Name;
  var valueObj = object[rootAttr];
  var result;

  classifyTree.listEntropy.forEach(function(item) {

    if (valueObj == item.value) {  // find out value
      if (typeof item.child != "undefined") {
        result = predictResultByTree(item.child, object);
      }
      else {  // finalResult exists
        result = item.finalResult;
      }
      return;
    }
  })

  return result;

}
```

4. Ví dụ minh họa

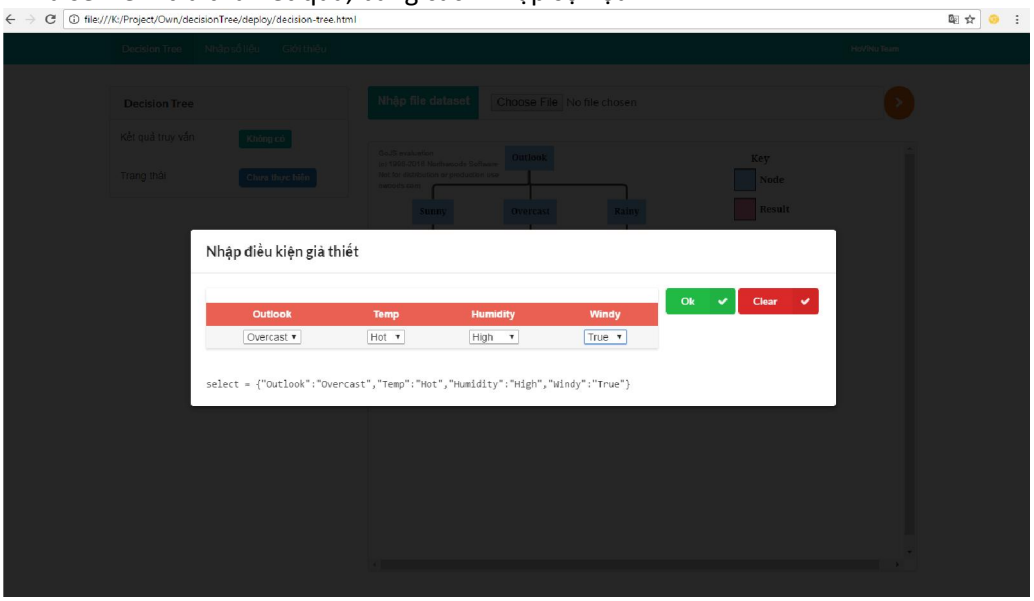
- Sau đây là trang giao diện chính của ứng dụng



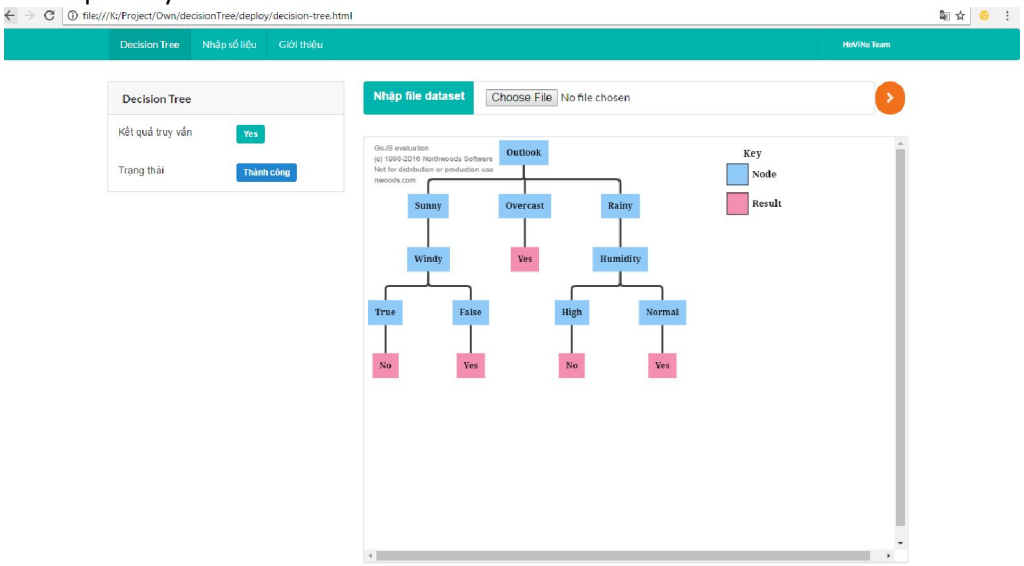
- Ta cần nhập file dataset, và ấn nút thực thi để vẽ cây. File dataset mẫu có đuôi là *.json trong thư mục “decisionTree\deploy\test1.json”

```
{
  "rawList": [
   ["Rainy", "Hot", "High", "False", "No"],
   ["Rainy", "Hot", "High", "True", "No"],
   ["Overcast", "Hot", "High", "False", "Yes"],
   ["Sunny", "Mild", "High", "False", "Yes"],
   ["Sunny", "Cool", "Normal", "False", "Yes"],
   ["Sunny", "Cool", "Normal", "True", "No"],
   ["Overcast", "Cool", "Normal", "True", "Yes"],
   ["Rainy", "Mild", "High", "False", "No"],
   ["Rainy", "Cool", "Normal", "False", "Yes"],
   ["Sunny", "Mild", "Normal", "False", "Yes"],
   ["Rainy", "Mild", "Normal", "True", "Yes"],
   ["Overcast", "Mild", "High", "True", "Yes"],
   ["Overcast", "Hot", "Normal", "False", "Yes"],
   ["Sunny", "Mild", "High", "True", "No"]
  ],
  "field": ["Outlook", "Temp", "Humidity", "Windy", "PlayGolf"],
  "listAttribute": [
   {
    "Name": "Outlook",
    "Values": ["Sunny", "Overcast", "Rainy"]
   },
   {
    "Name": "Temp",
    "Values": ["Hot", "Mild", "Cool"]
   },
   {
    "Name": "Humidity",
    "Values": ["High", "Normal"]
   },
   {
    "Name": "Windy",
    "Values": ["True", "False"]
   }
  ],
  "goalAttr": {
   "Name": "PlayGolf",
   "Values": ["Yes", "No"]
  }
```

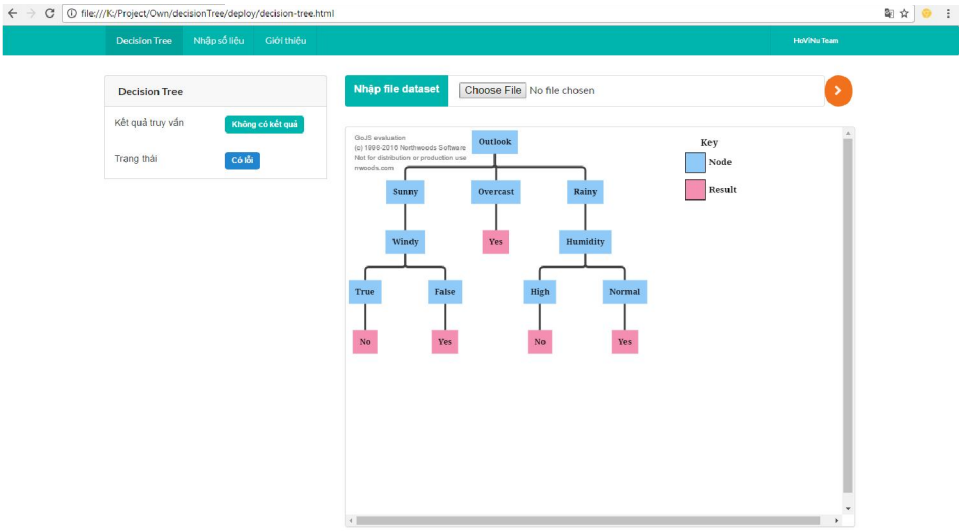
- Ta có “rawList” là thuộc tính chứa dữ liệu gốc, mỗi phần tử trong mảng sẽ tương ứng với mỗi hàng trong bảng. “Field” tên các thuộc tính theo thứ tự tương ứng với bản. Ngoài ra còn thuộc tính thông thường “listAttribute” và thuộc tính đích “goalAttr”.
- Ta sẽ kiểm tra thử kết quả, bằng cách nhập sộ liệu.



- Kết quả truy vấn sẽ là Yes.



- Trong tình huống nếu người dùng nhập không đủ dữ kiện, hoặc không tìm thấy kết quả. Trạng thái sẽ trả về lỗi. Do đó cần kiểm tra lại câu truy vấn.



5. Tổng kết

- Cây định danh giúp con người có thể mô phỏng được dữ liệu, tìm ra quy luật, mô hình hóa dữ liệu một cách dễ dàng và thuận tiện.
- Việc xây dựng cây định danh cũng khá đơn giản, và dễ dàng, giúp hỗ trợ và phát triển nhiều trong lĩnh vực máy học, data mining…

6. Hướng dẫn, source code và phân công

- Mở file “decisionTree\deploy\decision-tree.html” bằng trình duyệt Chrome, để hỗ trợ tốt nhất. Các bộ test mẫu nằm trong cùng thư mục này “test1.json , test2.json“ … File chính xử lý là “decision-tree.js”, và một số file khác để hỗ trợ.
- **Phân công:**
    - **Triệu Tráng Vinh**: đóng góp ý tưởng, code giải thuật, kiểm tra lỗi, tìm thư viên vẽ mô hình, viết báo cáo.
    - **Phan Đình Nguyên**: đóng góp ý tưởng, code giải thuật, kiểm tra lỗi, chỉnh sửa báo cáo.
    - **Trịnh Mẫn Hoàng**: đóng góp ý tưởng, code giải thuật, kiểm tra lỗi, chỉnh sửa báo cáo.

7. Tham khảo:

- http://www.saedsayad.com/decision_tree.htm
- http://gojs.net/latest/index.html