BRAC UNIVERSITY
Inspiring Excellence

Department of Computer Science and Engineering

Course Code:CSE422 **Credits: 1.5**

Course Name: Artificial Intelligence **Prerequisite:** CSE111, CSE221

Lab 08

Decision Tree Implementation

I. Lab Overview:

One of the first lessons in machine learning involves decision trees. The reason for the focus on decision trees is that they are simple, and at the same time, they provide reasonable accuracy on classification problems. In this tutorial, you'll learn:

a) What is a decision tree
b) How to construct a decision tree
c) Construct a decision tree using Python

II. Lesson Fit:

There is pre-requisite to this lab: CSE111, CSE221. You should have intensive Programming Knowledge and capability to understand algorithms.

III. Acceptance and Evaluation

Performed lab tasks will be evaluated by the Lab Instructor (LI)

a) Short viva will be conducted in each Lab or occasionally to examine your work.
b) You may work in groups but be aware that you will be evaluated individually; hence active participation during the Lab work demonstration is recommended.
c) There will be Lab handout after your work you have to handover it to LI

IV. Learning Outcome:

After this Lab, the students will be able to:

a) Understand basics of Decision tree for classification

b) Get an overview how to implement Decision tree

V. Activity Detail

a) Hour: 1-2

Getting Started:

a) Have a glance at Books "Python Data Science Handbook_ Essential Tools for Working with Data" by Jake VanderPlas, O'Reilly Media (2016)

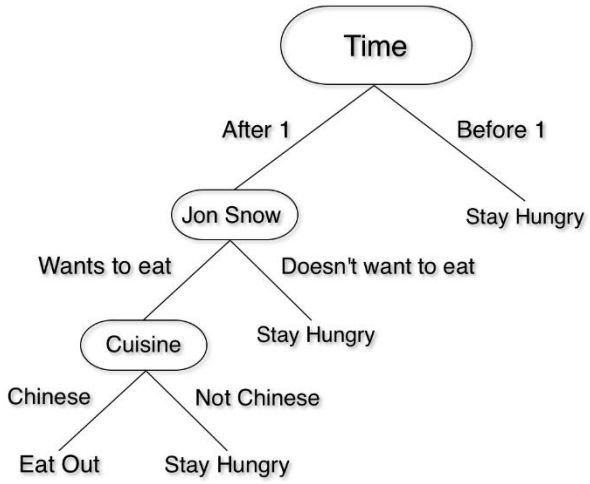
b) "Artificial Intelligence with Python written by Prateek Joshi, January 2017

c) Check \\TSR to see e-book copy, available datasets, codes, and tutorials

d) https://sefiks.com/2017/11/20/a-step-by-step-id3-decision-tree-example/

Discussion: What is a decision tree?

Let's skip the formal definition and think conceptually about Decision trees. Imagine you're sitting in your office and feeling hungry. You want to go out and eat, but lunch starts at 1 PM. You can think of your logic as a tree easily. If you want to go to lunch with your friend, Jon Snow, to a place that serves Chinese food, the logic can be summarized like the following tree:



Let's shift to the world of computers.

1. Start at the root node
2. Observe value of the attribute at the root

3. Follow the path that corresponds to the observed value
4. Repeat until we reach a leaf node, which will give us our decision

Discussion Continues: ID3 Algorithm

The most popular algorithm for decision trees is ID3, it is discussed in theory course. ID3 algorithm has to pick the attribute that best classifies the examples. How will it do that?

Information Gain and Entropy

One of the commonly used and beginner friendly ways to figure out the best attribute is information gain. It's calculated using another property called entropy. Let's see an example to make it clear:

You have 2 bags of full of chocolates. You discover the first bag has 50 chocolates. 25 of them are red and 25 are blue. Second bag also has 50 chocolates, all of them are blue. In this case, the first bag has entropy 1 as the chocolates are equally distributed. The second bag has entropy zero because there is no randomness. If you want to calculate the entropy of a system, we use this formula:

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

Here, c is the total number of classes or attributes and "pi" is number of examples belonging to the ith class. Let's try an example to clarify.

We have two classes, red(R) and blue(B). For the first box, we have 25 red chocolates. The total number of chocolates is 50. So pi becomes 25 divided by 50. Same goes for blue class. Plug those values into entropy equation and we get this:

$$Entropy(C) = -\frac{25}{50} log_2 \frac{25}{50} - \frac{25}{50} log_2 \frac{25}{50}$$

Solve the equation and you will get result 1. Calculate entropy for the second box, which has 50 red chocolates and 0 blue ones. You will get 0 entropy.

Information Gain

Information gain is simply the expected reduction in entropy caused by partitioning all our examples according to a given attribute. Mathematically, it's defined as:

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

S refers to the entire set of examples that we have. A is the attribute we want to partition or split. |S| is the number of examples and |Sv| is the number of examples for the current value of attribute A.

Building the Decision Tree

First, let's take our chocolate example and add a few extra details. We already know that the box 1 has 25 red chocolates and 25 blue ones. Now, we will also consider the brand of chocolates. Among red ones, 15 are Snickers and 10 are Kit Kats. In blue ones, 20 are Kit Kats and 5 are Snickers. Let's assume we only want to eat red Snickers. Here, red Snickers (15) become positive examples and everything else like blue Snickers and red Kit Kats are negative examples.

Now, the entropy of the dataset with respect to our classes (eat/not eat) is:

$$\begin{aligned} Entropy &= -\frac{15}{50} log_2 \frac{15}{50} - \frac{35}{50} log_2 \frac{35}{50} \\ &= 0.5210 + 0.3602 \\ &= 0.8812 \end{aligned}$$

Let's take a look back now—we have 50 chocolates. If we look at the attribute color, we have 25 red and 25 blue ones. If we look at the attribute brand, we have 20 Snickers and 30 Kit Kats. To build the tree, we need to pick one of these attributes for the root node. And we want to pick the one with the highest information gain. Let's calculate information gain for attributes to see the algorithm in action.

Information gain with respect to color would be:

$$\begin{aligned} Information\ Gain(Chocolates, Colors) = Entropy(Chocolates) \\ - (\frac{|red\ chocolates|}{|total\ chocolates|} \times Entropy(red\ chocolates)) \\ - (\frac{|blue\ chocolates|}{|total\ chocolates|} \times Entropy(blue\ chocolates)) \end{aligned}$$

We just calculated the entropy of chocolates with respect to class, which is 0.8812. For entropy of red chocolates, we want to eat 15 Snickers but not 10 Kit Kats. The entropy for red chocolates is:

$$\begin{aligned} Entropy(red\ chocolates) &= -\frac{15}{25}log_2\frac{15}{25} - \frac{10}{25}log_2\frac{10}{25} \\ &= 0.9709 \end{aligned}$$

For blue chocolates, we don't want to eat them at all. So entropy is 0.

Our information gain calculation now becomes:

$$\begin{aligned} Information\ Gain(Chocolates, Colors) &= 0.8812 - (\frac{25}{50} \times 0.9709) - (\frac{25}{50} \times 0) \\ &= 0.3958 \end{aligned}$$

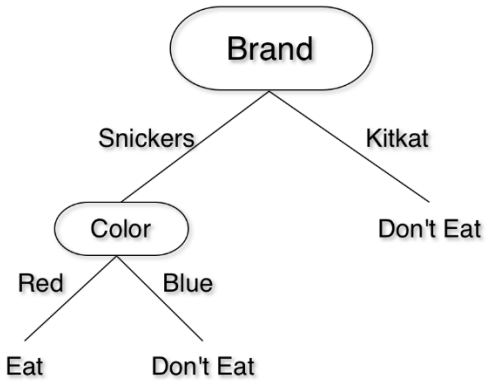
Let's look at the brand now. We want to eat 15 out of 20 Snickers. We don't want to eat any Kit Kats. The entropy for Snickers is:

$$\begin{aligned} Entropy(Snickers) &= -\frac{15}{20}log_2\frac{15}{20} - \frac{5}{20}log_2\frac{5}{20} \\ &= 0.8112 \end{aligned}$$

We don't want to eat Kit Kats at all, so Entropy is 0. Information gain:

$$\begin{aligned} Information\ Gain(Chocolates, Brand) &= 0.8812 - (\frac{20}{50} \times 0.8112) - (\frac{30}{50} \times 0) \\ &= 0.5567 \end{aligned}$$

Since information gain for brand is larger, we will split based on brand. For the next level, we only have color left. We can easily split based on color without having to do any calculations. Our decision tree will look like this:



You should have a solid intuition about how decision trees work now.

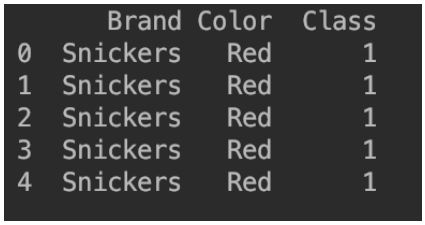
Implementing a Decision Tree with Python

Let's build a decision tree for chocolates dataset in \\TSR\...\CSE422 Lab\Lab04.

https://github.com/ishansharma/decision_trees_tutorial/

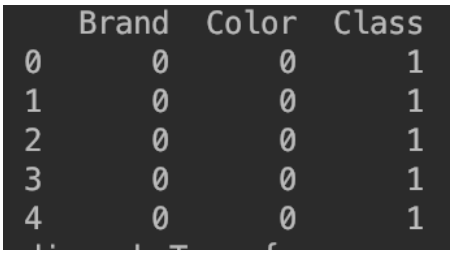
1) #Load the data using Pandas
   data = read_csv("data.csv")

   #You can take a quick look at loaded data using head() method in Pandas:
   print(data.head())
   #This will display the first 5 columns of our data.



	Brand	Color	Class
0	Snickers	Red	1
1	Snickers	Red	1
2	Snickers	Red	1
3	Snickers	Red	1
4	Snickers	Red	1

2) Class column decides to eat a chocolate or not. 1 means yes and 0 means no.
3) Scikit-Learn library doesn't support text labels by default, so we will use Pandas to convert our text labels to numbers. Simply add the following two lines:
   data['Color'] = data['Color'].map({'Red': 0, 'Blue': 1})
   data['Brand'] = data['Brand'].map({'Snickers': 0, 'Kit Kat': 1})
4) We just changed Color attribute to reflect 0 for Red and 1 for Blue. Similarly, we substituted 0 for Snickers and 1 for Kit Kat in the column Brand.
5) If you use head() to see the dataset, you'll see that brand and color values have changed to integers:



	Brand	Color	Class
0	0	0	1
1	0	0	1
2	0	0	1
3	0	0	1
4	0	0	1

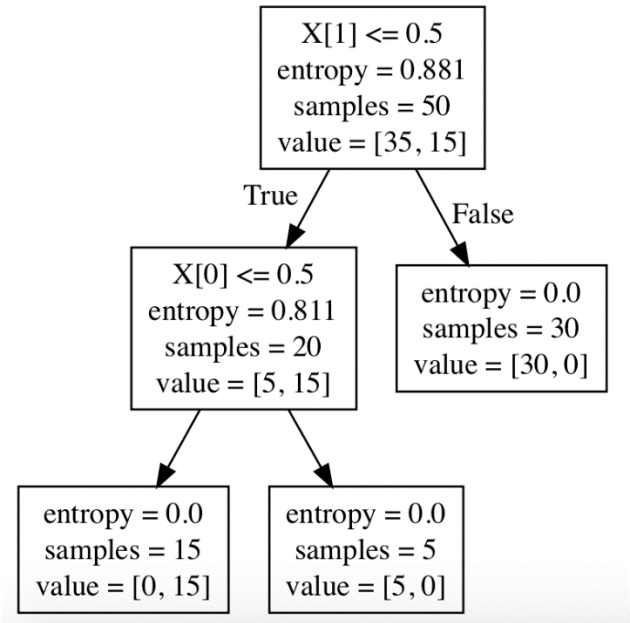
6) One last thing: it's a convention to denote our training attributes by X and output class by Y, so we will do that now:
   predictors = ['Color', 'Brand']
   X = data[predictors]
   Y = data.Class

Lab 05

7) Almost done. We're ready to train our decision tree now. Add the following two lines to train the tree on our data:
   decisionTreeClassifier = tree.DecisionTreeClassifier(criterion="entropy")
   dTree = decisionTreeClassifier.fit(X, Y)
8) Done? Let's quickly visualize the tree. Add these lines and run the program:
   dotData = tree.export_graphviz(dTree, out_file=None)
   print(dotData)

```
digraph Tree {
node [shape=box] ;
0 [label="X[1] <= 0.5\nentropy = 0.881\nsamples = 50\nvalue = [35, 15]"] ;
1 [label="X[0] <= 0.5\nentropy = 0.811\nsamples = 20\nvalue = [5, 15]"] ;
0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;
2 [label="entropy = 0.0\nsamples = 15\nvalue = [0, 15]"] ;
1 -> 2 ;
3 [label="entropy = 0.0\nsamples = 5\nvalue = [5, 0]"] ;
1 -> 3 ;
4 [label="entropy = 0.0\nsamples = 30\nvalue = [30, 0]"] ;
0 -> 4 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;
}
```

9) Copy this and head over to http://www.webgraphviz.com/. Paste the output and click "Generate Graph". You'll see a decision tree similar to one we made above:



10) This one is a bit harder to understand with all the extra information, but you can see that it split on column 1(Brand) first and then on column 1(Color).
11) Once you have learned the tree, future predictions are simple. Let's see if we want to eat a blue Kit Kat. Add following line to end of decision_tree.py file:

    print(dTree.predict([[1, 1]]))

12) The output will be [0] which means the classification is don't eat. If you try a red Snickers (print(dTree.predict([[0, 0]]))), output will be [1].

Activity List

(It is Not a Group Task, Try Individually)

Task 01: Mark 10 **Time:** 1 hour

Add a new attribute/column called "Taste" in your dataset, values will be bitter and Sweat. Let's say you want to have sweat chocolate only hence classify according to Taste attribute as well. You have to show whether splitting with taste attribute is a better option or not. Also Follow the process described in this tutorial above and generate results.

Hints: Use pandas library to solve this problem. First find out from web how to add new column in a CSV file using pandas. Then just follow the tutorial given straightforward.

Evaluation Process (VIVA): You have to explain your program and show your work to the Lab Instructor. Instructor may ask you some questions.