CSC <535/635> Data Mining

## Assignment-2 Report

### Submitted to:

### Dr. Jamil Saquer

### Author(s):

### Khaled Hossain

**ID3 Algorithm**

**Introduction**

For this assignment, I used two data sets, one was provided by the instructor and the other I downloaded from Kaggle. For the first part of the assignment I used the dataset provided by the instructor to implement the ID3 decision tree induction algorithm. The input was a list of tuples and each tuple represented one training sample. The training\_data was first converted into a data frame. The columns of the data were: - level, lang, tweets, phd, and hired. To solve the problem, I first by doing research and reading up about decision trees, looking at the PowerPoint slides, and watching the videos.

For the second part of the assignment I used a dataset from Kaggle which had 6 columns they are:- Id, age, income, student, and lastly credit\_rating. I chose this data set mainly because of two reasons firstly, it is simple. Secondly, this dataset will allow me to visualize the decision tree algorithm quickly and see if it is working. The most difficult part was creating the classify function.

**Background**

The main purpose of this assignment was to understand the ID3 algorithm and implement it using python to build a decision tree. The ID3 algorithm builds a decision tree by using training data. After that the resulting tree is used to classify future samples. The leaf nodes of the tree consist of class names and the non-leaf nodes are decision node. ID3 uses information gain to help it decide which attribute goes into a decision node. The ID3 algorithm uses information gain and entropy. Entropy is the measure of impurity, randomness, and uncertainty. The informationgain is based on the decrease in entropy after the dataset is split on the attribute. The classification function was used to predict the value of whether a new sample for our case a person would get hired or not. From the second dataset we can see whether a person buys a computer or not.

**Implementation**

The implementation of the algorithm was done by following a greedy approach by selecting an attribute that has the maximum information gain and minimum entropy. First, we calculate the entropy of the dataset than for each attribute we calculate the entropy for all categorical values, after that we calculate the information gain for the feature. Next we select the feature with the maximum information gain and repeat until we get the desired tree. I was unable to implement the handling of missing data.

***Text

Description automatically generated***

**Graphical user interface, text, application

Description automatically generated**

**Experimental Setup and Results**

The results for the experiment are as follows: -

Chart, scatter chart

Description automatically generated

Handling Missing data I was unable to implement this part:- A picture containing text

Description automatically generatedA picture containing text

Description automatically generated

Graphical user interface

Description automatically generated

By observing the results from the second data set we can see that if person is young, has a low income, and is a student but fair credit\_rating he will not buy a computer. However, if the person is older student, with a lower income he would still buy a computer.

**Conclusion (Extra Credit)**

This was a difficult assignment, but I was able to learn a lot of things. I learned about the decision trees, entropy, and information gain. There are a number of things that I can do to improve my code. I would start by implementing how to handle missing data and work on fixing my code so that I don’t have to build a decision tree every time I want to use a new dataset. Then I would start improving my algorithm. There are a number of things that I think would benefit this algorithm. Firstly, I would use more data to train my algorithm and have a larger dataset. I would implement a method to take into account missing and outlier values. I learned that the choosing of the attribute is very important and that the logarithm algorithm increases the time complexity of the calculation. Therefore, by using a different method to choose the attribute can be implemented. While I was doing research, I learned that it is possible to use the Taylor formula to simplify complex functions.**References**

Usman, M. (2018, June 21). Buy Computer. Retrieved October 07, 2020, from https://www.kaggle.com/usman27/buy-computer

**Code**

*You can use Notepad++ to copy your code right into your word document. Just* ***Ctrl-A****,* ***right-click*** *and select* ***Plugin Commands -> Copy text with Syntax Highlights.*** *Then in the word document:* ***right-click*** *and select* ***paste -> Keep Source Formatting****.**Otherwise just print out your code and staple it here (if asked to turn in a hard copy, which is not the case when the course is taught online).*

*Please follow common programming conventions in your code: use descriptive variable names, don’t overwrite built-ins or keywords, write your code in a modular format (use classes and functions), etc.*

*In the doc string at the top include your name, course code and name, and brief description of the assignment. If in a group: only one person should upload the report and code. Make sure to include the names of all group members.*

*When appropriate, please specify how we should test your code. Should it be run from IDLE, anaconda python prompt, as a notebook, …etc.*

"""

Program: HW2\_ID3.py

Programmed By: Khaled Hossain

Description: Homework - 2 the implementation of the ID3 algorithm

"""

#---------------------------------Imports--------------------------------------

#imports go here

**import** numpy **as** np

**import** math

#------------------------------------------------------------------------------

#---------------------------------Variables------------------------------------

#global variables go here; avoid if possible

**global** globalVar

moduleVar **=** "SomeValue"

#------------------------------------------------------------------------------

#---------------------------------Classes/Functions----------------------------

**class** **SomeClass():**

"""A descriptive statement about the class"""

**def** \_\_init\_\_**(**self**,** arg1**):**

#some descriptive comment about the method

self**.**arg **=** arg1

**def** \_\_str\_\_**(**self**):**

**return** str**(**self**.**arg**)**

**def** setArg**(**self**,** newArg**):**

self**.**arg **=** newArg

**def** getArg**(**self**):**

**return** self**.**arg

**def** someAlgorithmFunction**(**obj1**,** obj2**):**

#some descriptive comment

A **=** obj1**.**getArg**()** **+** obj2**.**getArg**()**

B **=** obj1**.**getArg**()** **-** obj2**.**getArg**()**

#some comment about the formula

result **=** **(**A**+**B**)/((**A**-**B**)\*(**A**+**B**))**

**return** result

#------------------------------------------------------------------------------

#---------------------------------Program Main---------------------------------

**def** main**():**

**if(**\_\_name\_\_ **==** "\_\_main\_\_"**):**

#some subsection

object1 **=** SomeClass**(**2**)**

object2 **=** SomeClass**(**10**)**

#some processing comment

algorithmResult = someAlgorithmFunction(object1, object2)

object1.setArg(algorithmResult) #some descriptive comment

object2.setArg(someAlgorithmFunction(object1, object2))

print("Object 1 content: ", object1)

print("Object 2 content: ", object2)

main()

#---------------------------------End of Program-------------------------------