CSE5AIF& CSE4002

Artificial Intelligence Fundamentals

2021 Assignment 2

Due Date: 11:59 PM (AEST/AEDT), Friday May 28, 2021

General Information

This assignment is to be done individually and contributes 25% of your final mark for this subject. The submission date for the assignment is 11:59 PM (AEST/AEDT), Friday May 28, 2021. Submission details are provided below. Make sure that you follow the directions carefully and that the files are named exactly as specified in the instructions.

The assignment is to be done individually. This means that answers to questions, and code that you write must be your own. You must not collude with other students in any way, and you must not outsource your work to any third party. For information on plagiarism, see the La Trobe University policy on academic misconduct at http://www.latrobe.edu.au/students/learning/academic-integrity.

Plagiarism is treated very seriously. Penalties will be applied and are strictly imposed.

Lab Objectives

The objective of this assignment is to utilize the **Scikit-Learn library** to implement the ID3 Decision Tree Machine Learning algorithm to create a classifier similar to that of Lab 8 with an alternative approach.

By the end of this assignment, you should have a better understanding of how Decision Trees work. Utilizing the trained tree visualization, you should have a reinforced

understanding of the core Decision Tree principles and how the trees split evaluations operate.

Background

In Lab 8, we created a predictive Decision Tree model from scratch; this implementation incrementally selected the best decisions to split on (evaluated based on the entropy principle) to provide an output classification based on our input data.

For this assignment, you will be utilizing some machine learning Python modules to create an ID3 predictive Decision Tree model, along with a visualization to better understand the classification process.

In order to compare the assignment implementation with that of Lab 8, you will be using the same classification scenario/dataset; that is, you will be creating a decision tree that will predict if a person is likely to go for a run based on the input environmental factors.

The models input parameters are:

- **outlook** (Forecast): Sunny, Overcast, or Rain
- **temp**: Cold, Mild, or Hot
- **humidity**: High, or Normal
- **wind**: Weak, or Strong

For training, you will be pairing these parameters with a **did_run** bool; this bool also acts as the models output value for testing/runtime predictions.

Implementation

Assignment Dependency Installation

You will have to install the following required dependency:

- scikit-learn
- pandas
- pydotplus
- IPython

Imports

You will be utilizing a number of well-known machine learning Python modules in this lab. These modules allow for ease of implementation of our decision tree and also include numerous learning tools to help boost your understanding.

Create the Dataset

The Pandas Python module is a very powerful, frequently use data analysis tool in all forms of machine learning; it allows you to very easily store and manipulate large datasets and is highly compatible/integrated with other machine learning tools/modules.
Unlike the Lab 8 implementation where we stored our data in Named Tuples, you will be storing your dataset into a Pandas dataframe.

A DataFrame is very similar to a Dictionary in standard Python; but has many additional useful features.

Initialize a pandas DataFrame as **input_df**; add our data into this dataframe by specifying the data keys and corresponding values.

Format the Training Data

Scikit-Learn is a powerful Machine Learning framework, you will be utilizing the included ***DecisionTreeClassifier*** class to create and train your Decision Tree.

The Scikit-Learn Decision Tree training function takes a binary representation of your data as input, before you can train your model, you must correctly format it. To do so you will have to convert your Pandas Dataframe to a one-hot representation.

One-Hot data, as the name suggests, involves categorically converting parameter values into a binary list representation where each bit represents the corresponding parameter value.

E.g.

For outlook = overcast we have:

Outlook = [overcast, sunny, rain]

Outlook = [1, 0, 0]

As you will have to employ a Pandas Dataframe, you can utilize the included **get_dummies** function to perform the needed conversion:

Train the Decision Tree

Once you have correctly formatted your data, you can move on to creating the Decision Tree. Create a new Scikit-Learn *DecisionTreeClassifier* named **classifier,** pass the "entropy" key as the criterion for the information gain.

Call the **fit** method on this classifier object to train the decision tree, inputting both your **one_hot_data** and corresponding **did_run** prediction training outputs.

Create a Graph Visualization

Next, utilize the **export_graphviz** method to generate the Dot Data representation of the trained decision tree graph.

The decision tree graph visualization will be saved in your working directory as a PDF named **output_graph.**

In one page, discuss the generated graph visualization in detail. You will have to submit the generated graph visualization (PDF- output_graph).

Test the Decision Tree

To test your developed model, you will have to pass the below input into your trained Decision Tree Classifier. Input this new data into the **predict** method of our **classifier_train** model.

Your test input is:
 # Outlook = sunny -> as one_hot: 0, 0, 1
 # Temperature = hot -> as one_hot: 0, 1, 0
 # Humidity = normal -> as one_hot: 0, 1
 # Wind = weak -> as one_hot: 1, 0

The output will now a True or False prediction; you will need to check that your implementation is working correctly by modifying these values and referencing the visualized tree to identify the correct result.

Marking Criteria

Your solution will be marked according to the following criteria:

- Discuss the generated graph visualization in detail and graph visualization.
- Implementing the Decision Tree using the given instructions.

Submission

Submission is electronic only and must be submitted to LMS. You must submit the following 2 files as a single ZIP file named your STUDENT ID:

- A word document or pdf,
- A file that contains all of your python code.

You must include your name and student ID in both of these files.