CS 422/622 Project 1: Decision Trees

Due 9/19/24 at 11:59pm

Logistics: This project includes four files, chatdt.py, test_data.csv, contrast_data.csv, and real_data.csv. The test data will work with the provided code out of the box, while the other two data files will require some work to get working properly. This project uses ChatGPT in its construction, and the coding portion of the project doesn't prohibit the use of ChatGPT, but it is highly recommended that you attempt to solve the problems on your own first. The written portion of the project should not use any generative AI, and is required to be done on your own. To run the code with the different data files you will provide a command line argument. This will look like **python3 chatdt.py 1**. 1 is test_data.csv, 2 is contrast_data.csv, and 3 is real_data.csv.

Deliverables: You should submit a single ZIP file, containing your project code (chatdt.py files), the data files (∗.csv) and your writeup (PDF). Grad students are also expected to include their LaTeX source files (∗.tex). Your zip file should be named lastname_firstname_project1.zip. For example, a zip file for Sara Smith would be smith_sara_project1.zip. Your code should run without errors on the ECC linux machines. If your code does not run for a particular problem, you will lose 50% on that problem.

1 ChatGPT Decision Tree Code (60 Points)

File name: **chatdt.py**

A file is provided to you that uses code generated from ChatGPT. It was asked to generate a decision tree algorithm from scratch using entropy, and to only accept numpy arrays. Your task is to go through this code and fix a series of errors that ChatGPT failed to account for. These errors will be listed for you, and each fix will be worth 20 points. The idea is to learn how to read code, and get you a little bit more familiar with the way ChatGPT approaches code generation, as well as how to address areas where ChatGPT can fail. Some of these errors can be potentially solved using ChatGPT, but the expectation is for you to attempt to solve them on your own in order to learn.

Error 1: Max Depth (20 Points)
The provided code fails to properly build a tree when no max depth is provided as an argument (eg. max_depth = None). It also does not properly terminate the tree if 100% accuracy is achieved before reaching a max depth. The expected fix is to have the tree terminate if 100% accuracy is achieved, regardless of what the max depth was set to (for example, if max_depth = 10 but the tree achieved perfect accuracy on depth 3, then terminate at depth 3). Also have the tree terminate in some way if no max depth is provided.

Error 2: Contradictory data samples (20 points)
The contrast_data.csv contains duplicate samples that give opposite labels, as this can sometimes happen in real world data sets. This causes the decision tree to give an error and prevents the code from running. Find a way to have the decision tree accept and run with the contradictory data samples, without changing the data. If this is done successfully the tree will be built, but the final prediction accuracy should go down.

Error 3: No Accuracy Method (20 points)
The code failed to generate a method that allows us as programmers to see how well the model is doing. Implement an accuracy method that measures how many samples the tree is accurately predicting, and returns that ratio as a percentage. Of the three datasets provided, only the test_data.csv should give 100% accuracy when trained with a low max depth.

Grad Students — Error 4: Train and Test Data (20 points)
Real world datasets require some way to properly validate the model. This is usually done by splitting the data into training data, and testing data. These splits allow us to train the model on the training data, then validate its performance against the testing data. Implement a method that splits the data into a training set, and a testing set. The fit function should only be passed the training set, while the pred function should be given the testing set. The function should have one parameter that is the amount by which to split the data as measured by how big the test set will be. For example, if 0.15 is passed, then the 85% of the data will

be training, and 15% will be testing. The data that is split should be randomly sampled. The real_data.csv is provided with enough data samples to properly test this function.

Grading: Your code will be ran in the latest Ubuntu environment as provided. Your code is expected to run with at least the test_data.csv without issues, as that should work as provided. If the code fails to compile, it will be a 50% penalty. Your code will be ran as is with the same three data files that are given.

2 Write Up (40 Points)

File name: **report.pdf**

Create a write up detailing what each individual method in the DecisionTree and the DecisionTreeNode class is doing. Each method should have at least a couple of sentences summarizing either what the algorithm is doing, or how it is used by the algorithm. You do not need to describe what the cmd line arguments function is doing. That is there simply for loading the data. DO NOT USE ChatGPT for the write up. The report is about detailing your understanding of the algorithm. The report should be saved as a pdf and uploaded with the rest of your code.

Grad Students — LaTeX Files:
Grad students are expected to write their report in LaTeX and include the source .tex file with their pdf upload. If you are in need of a resource to write LaTeX files, overleaf.com is an excellent free resource available.