Intelligent Systems Engineering
ECE 449
Fall 2023

Laboratory Exercise #5

Resources

R. Kruse et al., *Computational Intelligence: A Methodological Introduction, 3rd Ed.* (text)

- Chapter 11, sections 11.1-11.3
- Chapter 12
- Chater 13.1 Genetic Algorithms
- Chapter 21.2 Evolutionary Fuzzy Algorithms

ECE 449 Slides from last year (Dr. Musilek)

- Evolutionary Computation
- Evolutionary Optimization
- Genetic Algorithms

skfuzzy Documentation

- Examples https://pythonhosted.org/scikit-fuzzy/auto_examples/index.html

EasyGA Documentation

- https://github.com/danielwilczak101/EasyGA

Summary

Genetic-fuzzy systems are a type of machine learning algorithm that combines the flexibility of machine learning with the interpretability of fuzzy inferential systems. One of the ways in which such systems can be built from a dataset is, first, to declaratively specify an initial fuzzy system, and then refine the fuzzy sets underlying the rules using genetic algorithms. This is the approach we shall pursue in this exercise. A small dataset on tip amounts given observed service attributes has been provided; you will use genetic algorithms to refine the fuzzy tree from Lab #4 to accurately model this dataset.

Discussion

As an interpretable AI model, fuzzy systems have always been interesting to researchers and developers of intelligent systems. We've seen the black-box problem in neural networks, and how it undermines trust in AI systems in numerous ways. It's thus no surprise that interpretable models are desirable. However, fuzzy systems have a major drawback: they can't, on their own, be adapted to fit a dataset more accurately. The expert knowledge that fuzzy systems are based on is human understanding, which copes with the complexity of the real world by abstraction. The specifics of a particular situation are not so easily captured, even if humans are able to adapt and react correctly in the heat of the moment. Thus, fuzzy systems tend to be less accurate than other AIs, which is a barrier to their wider adoption.

One of the responses to this problem, back in the late 1980s, was to start looking at how hybrids of fuzzy logic and machine learning might be created. The earliest attempts focused on neuro-fuzzy systems, hybrids of neural networks and fuzzy logic. Research in that direction is still ongoing, these days focusing on deep neuro-fuzzy systems; Dr. Dick's research is also on this topic. However, a rival approach that began appearing in the mid-1990s is to optimize fuzzy systems (perhaps the entire rulebase, perhaps only the fuzzy sets for each term, etc.) using genetic algorithms. These so-called fuzzy systems have been found to be the most effective way to induce a fuzzy rulebase from data (you'll notice that fuzzy rulebases are still the central component of genetic-fuzzy systems).

In this lab exercise, you are going to expand on the fuzzy tree for recommending tip amounts that you built in Lab #4. You will be provided a small dataset that associates observed food and service quality attributes with a tip amount; you are going to design a genetic fuzzy system that modifies your fuzzy tree to model this dataset as accurately as possible. This dataset was in part created by adding random noise to attribute values in some records; this means that the data points sometimes fall out of the allowed range [0,1] for the input variables. Your pipeline should treat this as a data cleaning problem, and clip values below 0 to exactly 0, with values above 1 set to exactly 1.

We will use the EasyGA library in Python for our genetic algorithms; at this point it appears to be the best freely-available choice. As we already have a basic tree structure, and rules for each rulebase within it, we will confine our attention to optimizing the fuzzy sets for each quality attribute, the inputs to the final tipping rulebase, and the intermediate and final output variables (i.e. all the fuzzy sets in your tree). The resulting algorithm will be what we usually call a genetic fuzzy tree.

Requirements
The tipping dataset is already divided into a training and out-of-sample testing dataset. Use the EasyGA and skfuzzy libraries to build a genetic fuzzy tree as described above that optimizes your fuzzy tree from Lab #4 to accurately model the tipping dataset. This model should, at a minimum, outperform the un-optimized fuzzy tree from Lab #4 on the test dataset.

Submission
All of the above should be automatically performed in a Python-based pipeline, and submitted as a Jupyter notebook. It is NOT necessary that the pipeline() object from scikit-learn be used in this assignment, although it is not prohibited. Submit the notebooks via eClass in your lab section before the deadline.

Grading

Genetic fuzzy tree constructed and trained	25%
Test dataset correctly executed	50%
Genetic fuzzy tree outperforms tree from Lab #4 (on the test dataset)	25%