CSCI 447 — Machine Learning: Soft Computing

Project #1

Assigned: August 26, 2019
Project Due: September 13, 2019

The purpose of this assignment is to provide a gentle introduction to the field of machine learning by having you implement one algorithm and test it on a few real-world data sets. For purposes of this assignment, we won't tell you exactly what the name of the algorithm is, or provide it's theoretical basis, until later. You'll just have to trust us.

This experiment requires you to download five data sets from the UCI Machine Learning repository and train a classifier for each of these data sets. The data sets you will use are the following:

1. Breast Cancer:

   `https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+\%28Original\%29`

   This breast cancer databases was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg.

2. Glass:

   `https://archive.ics.uci.edu/ml/datasets/Glass+Identification`

   The study of classification of types of glass was motivated by criminological investigation.

3. Iris:

   `https://archive.ics.uci.edu/ml/datasets/Iris`

   The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant.

4. Soybean (small):

   `https://archive.ics.uci.edu/ml/datasets/Soybean+\%28Small\%29`

   A small subset of the original soybean database.

5. Vote:

   `https://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records`

   This data set includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the Congressional Quarterly Almanac.

When using these data sets, be careful of some issues.

1. Some of the data sets have missing attribute values. When this occurs in low numbers, you may simply edit the corresponding values out of the data sets. For more occurrences, you should do some kind of "data imputation" where, basically, you generate a value of some kind. This can be purely random, or it can be sampled according to the conditional probability of the values occurring, given the underlying class for that example. The choice is yours, but be sure to document your choice.

2. Most of attributes in the various data sets are either multi-value discrete (categorical) or real-valued. You will need to deal with the continuous valued attributes in some way. Specifically, you will need to discretize them in some way for both algorithms and then proceed as in the multi-valued categorical case. It's up to you how you do that, but again, please document your approach.

Here is the algorithm you are to implement. These instructions apply to a single training data set. For this, we need some notation. We will use #{pred} to represent that we are counting the number of times the predicate given by "pred" is matched.

1. For each class in the training set, calculate

$$Q(C = c_i) = \frac{\#\{\mathbf{x} \in c_i\}}{N}$$

   In other words, for each class, divide the number of examples in that class by the total number of examples N in the training set.

2. Logically, you will then separate the data into their respective classes. In reality, you don't need to do that, but it makes the explanation easier. So now think of each class having its own subset of data. We will then consider each class in turn.

3. For each attribute A_j in the class-specific training set, calculate

$$F(A_j = a_k, C = c_i) = \frac{\#\{(\mathbf{x}_{A_j} = a_k) \wedge (\mathbf{x} \in c_i)\} + 1}{N_{c_i} + d}$$

   where d is the number of attributes and $N_{c_i} = \#\{\mathbf{x} \in c_i\}$. In other words, for each attribute value, divide the number of examples that match that attribute value (plus one) by the number of examples in the class (plus d).

That's the entire training algorithm. Then to classify an example from the test set, do the following for each class. Calculate only for the attribute values a_k that exist in the example

$$C(\mathbf{x}) = Q(C = c_i) \times \prod_{j=1}^{d} F(A_j = a_k, C = c_i)$$

Then return

$$class(\mathbf{x}) = \underset{c_i \in C}{\operatorname{argmax}}\ C(\mathbf{x}).$$

In other words, return the class with the highest value for $C(\mathbf{x})$.

You should complete the following steps for this assignment:

- Download the five (5) data sets from the UCI Machine Learning repository. You can find this repository at `http://archive.ics.uci.edu/ml/`.
- Pre-process the data to ensure you are working with complete examples (i.e., no missing attribute values) and discrete features only.
- Implement the above algorithm and test it on two different versions of the data. The first version is what you get from the repository without change (other than, possibly, data imputation). The second version goes through your data, selects 10% of the features at random and shuffles the values within each feature, thus introducing noise into the data.
- Select and implement at least two (2) different evaluation measures (i.e., loss functions) that you will use to evaluate your algorithm.
- Develop a hypothesis for each data set based on expected performance on the different data sets.
- Design and execute experiments using 10-fold cross-validation to test your hypotheses, comparing performance on the unaltered and altered data sets from the UCI repository.
- Write a paper that incorporates the following elements, summarizing the results of your experiments. Make sure you explain the experimental setup, the tuning process, and the final parameters used for each algorithm.
  1. Title and author name(s)
  2. A brief, one paragraph abstract summarizing the results of the experiments

3. Problem statement, including hypothesis
4. Description of algorithm implemented
5. Description of your experimental approach
6. Presentation of the results of your experiments (in words, tables, and graphs)
7. A discussion of the behavior of your algorithms, combined with any conclusions you can draw
8. Summary
9. References (You should have at least one reference related to the algorithms implemented since you will know what it is by the time the assignment is due. You should also include a reference to the data sources, and any other references you consider to be relevant. You may not reference websites except for the data sets.)

- Create a video that is no longer than 5 minutes long demonstrating the functioning of your code. This video should focus on behavior and not on walking through the code. You need to show input, data structure, and output. How you do this is entirely up to you, but be sure it will convince the grader that your program works.
- Submit your fully documented code, results of the runs of your algorithm, your paper, and a video demonstrating the code functioning.