

Assignment 5

Part I: Questions (20 points each)

1. Compute the empirical distribution of the three types of irises in the iris dataset (see part II). Using the empirical distribution, compute the entropy of the iris species.
2. Regularization is very likely to result in a learner not learning the training data as well, yet it is an important component of machine learning algorithms. What is regularization and why is it important?
3. Assume the following knowledge base:

$nutrients \wedge water \wedge sun \Rightarrow flower$

$flower \wedge pollinator \Rightarrow fruit$

$bat \vee bird \vee bee \Rightarrow pollinator$

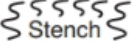
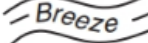



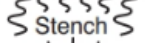


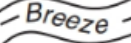
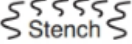


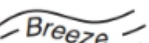

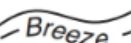
$irrigation \Rightarrow water$

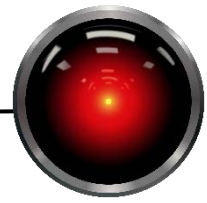
and the current set of facts:

$irrigation \wedge nutrients \wedge sun \wedge bat$

Without using the resolution rule, prove that fruit will be borne.

4. Given the wumpus world of Figure 7.2 from your book where position (x, y) denotes the cave in column x, row y (e.g. wumpus is at 1,3):

4	 Stench		 Breeze	
3		 Breeze  Stench  Gold		 Breeze
2	 Stench		 Breeze	
1	 START	 Breeze		 Breeze
	1	2	3	4



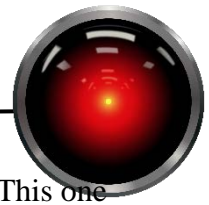
- a. What are the percepts for an agent at 2,3?
 - b. Write a set of logical sentences (see section 7.4.3) that describe the predicates that can be inferred by percepts at 2,3. Do not include knowledge from any other position that could infer more details.
5. In addition to what we know about the starting conditions of a wumpus world, assume that a stench has been observed at locations (2,1) and (1,2). Write the knowledge base in conjunctive normal form and show whether or not the question $W_{2,2}$ (there is a wumpus at location 2,2) is entailed by the knowledge base using the resolution rule.

Part II: Programming Assignment – 100 points

The code package for this assignment on Blackboard contains a mostly complete implementation of several machine learning algorithms. These algorithms are not implemented for speed and are only effective for toy problem sets. If you want to use real-world machine learning problems after the course, there are more efficient libraries such as [scikit-learn](https://scikit-learn.org/) which you will not be using here.

In module `ml_lib.learning`, you will find the following classes along with many classes that you do not need to worry about but are welcome to investigate:

- `DataSet` – Read a data set. Given the name argument, it will find the CSV data files that are located in `ml_lib/aima_data`. Each data set has two files with different extensions: `.txt` – a description of the problem and data, `.csv` – the data in comma separated value format. There are several datasets available:
 - `abalone` – 4,177 examples of attribute data to predict an abalone's age from measurements taken from harvested (killed) abalone.
 - `iris` – 150 sets of measurements of characteristics of three species of iris plants, task is to predict the species: *Iris Setosa*, *Iris Versicolour*, or *Iris Virginica*.
 - `orings` – Data from 23 NASA space shuttle launches prior to the Challenger crash. The task is to predict the number of o-rings that are likely to exhibit thermal stress. Please note that unlike the other datasets, the attribute to predict is not the last one and you must use `DataSet`'s `setproblem` method identify the attribute to predict before calling the learning algorithm. You might want to examine whether or not the mission number is relevant (not required for assignment).
 - `restaurant` – Professor Russell's restaurant waiting rules (no `.txt` file provided as the problem is explained in the book).
 - `zoo` – 101 animals with numerical and boolean attributes, much like our flies/no flies example in class. Task is to predict broad classes of animals: bird, fish, reptile, mammal, insect, amphibian, or shellfish.
- `DecisionTreeLearner` – Given a dataset, build a decision tree.
- `NeuralNetLearner` – Given a dataset, build a neural net. See caveats about the neural nets algorithm in the specification of `driver.py`



- `cross_validation` – Non-standard implementation of cross-validation. This one shuffles before each fold. Takes a handle to a learning function, a `DataSet`, the number of folds `k`, and the number of times the cross validation is to be repeated.

Write the following:

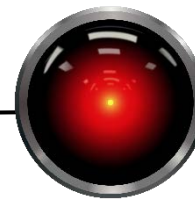
- `std_cv.py` – Write a standard `cross_validation` routine that has a similar interface to `ml_lib.learner.cross_validation` (feel free to reuse code). It differs as follows:
 - Return values are the mean error rate, the standard deviation of the error rate, a list containing the error rate for each fold, and a list of models trained. Unlike the `ml_lib` package implementation, results on the training set are neither generated or returned (these usually aren't all that useful other than as a sanity check).
 - *Data are not shuffled*, resulting in every example being used $N-1$ times for training and 1 time for test.
 - There is no `trials` parameter as running the same data multiple times would produce the same result.
- `driver.py` – Write the following functions:
 - `shuffle_data` – Shuffles the example list.
 - `learn` – Given a dataset, create a shuffled copy of the data. Run a 10 fold cross validation (`std_cv.cross_validation`) using decision trees and neural nets (`DecisionTreeLearner` and `NeuralNetLearner`). The data are only shuffled once, so that `std_cv.cross_validation` will use the same exact training and test data on each of the folds for both learners.

Use the default options for `NeuralNetLearner`, but you are welcome to explore if you are curious. Hint: The neural net implementation given does not work well on categorical data, `DataSet.attributes_to_numbers` will convert all categorical data to numbers. Note that this is not always the best way to do this, sometimes we use so-called “one hot” vectors, a binary vector the length of the category list with a single bit set to one. There are some advantages to this that are beyond the scope of this assignment.

Note that the `abilone` dataset is significantly longer than the other data sets, and you may omit running that one if you wish, although successful execution of `abilone` will earn you an additional 5 points up to the maximum score for the assignment.

To turn in:

- Soft and hard copies of code
- Output showing the mean, standard deviation, and list of error rates for each test set and each learning method. Example output for the `iris` data set can be seen on the next page (numbers may vary as data have been randomly shuffled)



Sample output for one of the datasets.

Mean	StdDev	Errors for each fold	Corpus	Learner
0.060	0.106	0.067,0.133,0.333,0.000,0.067,0.000,0.000,0.000,0.000,0.000	iris/75	DecisionTreeLearner
0.255	0.275	0.429,0.500,0.429,0.625,0.571,0.000,0.000,0.000,0.000,0.000	iris/75	NeuralNetLearner