# CSE 5800 Mining/Learning and the Internet—HW2
# Due Sep 30, Wed, 6:30pm
# Submit Server: course=ml-internet , project=hw2

Implement and evaluate LERAD (LEarning Rules for Anomaly Detection). Do not generate "wildcard rules" in Step 1 (in the paper) since they get relatively high scores in small data sets.

1. Allow parameters:
   (a) number of pairs of examples for generating candidate rules ($L$ in the paper)
   (b) maximum number of rules per pair of examples ($M$ in the paper)
   (c) number of examples in the sample set ($|S|$ in the paper)
   (d) number of examples in the validation set as a percentage of the entire training set [e.g. 10% means 90% for training, 10% for validation]
2. Vary the score threshold, report AUC (area under curve) upto 1%, 10%, and 100% false alarm rate.
3. Three data sets:
   (a) toy data set on the course web site
   (b) intrusion detection on the course web site
   (c) your own data set
4. A report (in pdf) that discusses the following:
   (a) Sensitivity analysis of parameters: for the second data set,
      i. vary each of the four parameters (keeping the other three constant),
      ii. calculate AUC upto 1% false alarm rate,
      iii. plot AUC vs. value of a parameter,
      iv. discuss the value for each parameter that seems to achieve the highest AUC.
   (b) Robustness to noise (errors/attacks in the "normal" training data): for the second data set,
      i. add 1% upto 10% (1% increment) attacks into the training data,
      ii. calculate AUC upto 1% false alarm rate,
      iii. plot AUC vs. noise (0% to 10%)
5. Implementation:
   (a) preferrably use one of these programming languages: C, C++, Java, Python, or LISP.
   (b) input files: attributes description, training data, test data
   (c) suggestion: three potential modules:
      i. Miner/learner: input training examples/instances, output a ruleset
      ii. Detector/predictor: input the ruleset and labeled instances, output the classifications/predictions and AUC.
      iii. Ruleset printer: if the output from the learner is human-readable, no need for a ruleset printer; otherwise, build a ruleset printer so that we can see the rules.
6. Submission:
   (a) source code
   (b) your data set
   (c) report in pdf
   (d) README.txt (how to compile and run your program/experiments on code.fit.edu or hopper.cs.fit.edu)