CS 5033 - Machine Learning: Homework 6

Spring 2022

Due: Sunday, May 1, 2022 (End of day)

We are coming back to the dataset that we used in homework 5. Namely, we will be using a UCI simulated electrical grid stability data set that is available here:

https://archive.ics.uci.edu/ml/datasets/Electrical+Grid+Stability+Simulated+Data+.

This dataset has 10,000 examples using 11 real-valued attributes, with a binary target (stable vs. unstable). The target value that you are predicting is the last column in the dataset.

Remark 1 (Cross-Entropy). For the cross-entropy values that you want to report in the questions below, please use the following formula (empirical risk using the cross-entropy loss):

$$\hat{R}_S(h, c) = -\frac{1}{m} \sum_{i=1}^{m} \left[y_i \ln(p_i) + (1 - y_i) \ln(1 - p_i) \right] ,$$

where

$$\begin{cases} m & \text{is the size of the sample } S \text{ where we evaluate our hypothesis } h, \\ y_i \in \{0, 1\} & \text{is the true label } c(x_i) \text{ of the instance } x_i, \text{ and} \\ p_i & \text{is the probability of assigning the positive label to instance } x_i \text{ by our hypothesis.} \end{cases}$$

Exercise 1 – Preprocessing (10 points). You have already done this part in homework 4. However, since you may need to refresh your memory with what you did, this part is worth a few points.

(a) Remove columns 5 and 13 (labeled p1 and stab); p1 is non-predictive and stab is target column that is exactly correlated with the binary target you are trying to predict (if this column is negative, the system is stable).

(b) Change the target variable to a number. If the value is stable, change it to 1, and if the value is unstable, change it to 0.

(c) Remove 20% of the examples and keep them for testing. You may assume that all examples are independent, so it does not matter which 20% you remove. However, the testing data should not be used until after a model has been selected.

(d) Split the remaining examples into training (75%) and validation (25%). Thus, you will train with 60% of the full dataset (75% of 80%) and validate with 20% of the full dataset (25% of 80%).

Exercise 2 – Artificial Neural Network (20 points). You may use `sklearn.neural_network.MLPClassifier`.

(a) Fit an artificial neural network to the training data using 1 hidden layer of 20 units as well as another neural network that has 2 hidden layers of 10 units each.

(b) For each model made in (a), make a probabilistic prediction for each validation example. Report the cross-entropies between the predictions and the true labels in your writeup.

(c) Which neural network performs the best on the validation data? Report this in your writeup. Train a new neural network using the architecture that performed better among the two using the training and validation data. Make a probabilistic prediction for each testing example using this model and save them for later.

Exercise 3 – Decision Trees (20 points). For this problem you can use the scikit-learn method `sklearn.tree.DecisionTreeClassifier`.

(a) Fit a decision tree to the training data using the Gini impurity index and max tree depth of 5.

(b) Using the model created in part (a) make a probabilistic prediction for each validation example. What is the cross-entropy on these predictions and the true labels? Put this value in your writeup.

(c) Fit a decision tree to the training data using information gain and max tree depth of 5.

(d) Using the model created in part (c) make a probabilistic prediction for each validation example. What is the cross-entropy on these predictions and the true labels? Put this value in your writeup.

(e) Which model performed better on the validation data? Report this in your writeup. Train a new decision tree on the training and validation data using whichever measure created the best model in (a)-(d), with a max tree depth of 5. Make a probabilistic prediction for each testing example and save them for later.

Exercise 4 – Boosting (20 points). For this problem you may use `sklearn.ensemble.AdaBoostClassifier`.

(a) Fit boosted decision stumps (max tree depth of 1) to the training data allowing at most 20, 40, and 60 decision stumps (base estimators) in each model.

(b) For each model trained in (a), make a probabilisitc prediction for each validation example. Report the cross-entropies between the predictions and the true labels in your writeup.

(c) Which upper bound on the number of allowed base classifiers generates the best performing model? Report this in your writeup. Train a new AdaBoost classifier using this bound on the number of maximum allowed base classifiers, using the training and validation data. Make a probabilistic prediction for each testing example using this model and save them for later.

Exercise 5 – ROC Curve (30 points). For this exercise **you must write your own code; no scikit-learn, except maybe to compute AUC**.
For each model produced in Exercises 2-4 do the following:

(a) Determinize the testing predictions made above, using 1001 different probability thresholds (0.000, 0.001, 0.002, . . ., 0.999, 1.000). "Determinization" means converting the probability to a deterministic class label (0 or 1). Use (1) below for determinization. We have that p^* is the critical threshold; p_i is the predicted probability for example i; and P_i is the resulting deterministic prediction:

$$P_i = \begin{cases} 1, & \text{if } p_i \geq p^* \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

(b) At each of the 1001 probability thresholds, compute the true positive rate (TPR) and false positive rate (FPR). Recall that these values are easily computed from the confusion matrix. (You would have to re-calculate the confusion matrix for each one of these thresholds, for each model.)

(c) Plot the ROC (receiver operating characteristic) curve, using the 1001 points created in part 6b. If you have forgotten what a ROC curve looks like, see our notes on model evaluation. The ROC curve **must** contain a point at the bottom left (0, 0) and top right (1, 1). Also, it must contain the dashed grey line, indicating the performance of a random predictor. Include the ROC curve for each model in your write-up.

(d) Find the probability threshold yielding the highest Youden index (TPR - FPR). Report the Youden index and the corresponding probability threshold for each model.

(e) Compute the AUC (area under the curve) for each model. You may use the function `sklearn.metrics.roc_auc_score` for this part.