Homework 2, Due April 18

MPCS 53111 Machine Learning, Winter 2017
University of Chicago

Practice problems, do not submit

1. **Decision lists.** Exercises 18.13 and 18.14 from Russell-Norvig.

Graded problems, submit

2. **χ^2-pruning.** Exercise 18.8 from Russell-Norvig.

3. **PAC learning.** Consider a "boolean classification" problem in which the output is boolean and the inputs are integers. Further, restrict the hypotheses to the form

$$y(\mathbf{x}) = (a \leq x_1 \leq b) \text{ and } (c \leq x_2 \leq d),$$

in which x_1, x_2 are inputs; y is the output; and a, b, c, and d are integers in $1, \ldots, n$, where n is some given constant. (E.g., if $n = 100$, then one hypothesis is $(90 \leq x_1 \leq 100)$ and $(15 \leq x_2 \leq 15)$, which is true for (95, 15) and false for (95, 16).)

   (a) Show that the number of hypotheses is $(n(n+1)/2)^2$. (Hint: The number of hypotheses is equal to the number of distinct rectangles on the x_1-x_2 plane with boundaries restricted to integer positions from 1 to n. Show how to count such rectangles.)

   (b) How many training examples are sufficient to assure that any hypotheses with zero training error will have a generalization error of at most 10% with probability 95%.

4. **Measuring similarity between two classifications.** In this problem, we will implement the quadratic weighted kappa, a measure of similarity between two different classifications into an ordered categorical variable. This measure is found by constructing three $k \times k$ matrices, where k is the number of categories: O, E, and W.

   - The entries o_{ij} of O are the observed proportion of observations that are classified as i by the first classifier and j by the second.

- The entries e_{ij} of E are the proportion of observations that are classified as i by the first classifier, times the number of observations that are classified as j by the second classifier. (These entries denote the *expected* proportion of joint classifications that would occur by chance.)
- The entries w_{ij} of W are the weights, which we define here to be $(i-j)^2$.

Then the quadratic weighted kappa is given by $\kappa = 1 - \frac{\sum w_{ij} o_{ij}}{\sum w_{ij} e_{ij}}$. See Wikipedia for further details.

Implement a function `qwk(y1, y2)` that takes two classifications `y1` and `y2`, and outputs their quadratic weighted kappa. Then compare your implementation against scikit-learn's implementation with random vectors to make sure yours is correct.

5. **Risk Classification for Prudential.** Here we'll explore the Prudential problem at Kaggle. This will reinforce the concepts of training, testing, crossvalidation, bias, variance, etc. Equally importantly, it will help you become familiar with `scikit-learn`, the most popular machine learning library in Python. You will also get practice using `numpy` and `pandas`. In the following, when a function from either of the libraries is mentioned, please read the corresponding documentation to familiarize yourself with the related concepts and the input parameters. Whenever you need to score a model, use **quadratic weighted kappa.**.

   (a) Load the training data from the `train.csv` file at the site. For the remaining questions use a small enough dataset so that the plots can be constructed in a few minutes, such as a sample of 10,000 rows. Specify the number of rows you used clearly in the discussion section. (While you are developing and testing your scripts, use something even smaller.) For this you may want to use `DataFrame.sample`.

   (b) Several of the variables have missing values. Use `Imputer` to fill in the missing values. Choose an imputation strategy (mean, median, or most frequent) depending on the type of feature (categorical or continuous). Briefly describe your choices.

Let's say feature X_k has missing values, which you have imputed. Could it be useful to add an additional boolean feature $X_{k'}$ that indicates which of the rows had missing values for X_k? Explain your answer.

(c) `scikit-learn` does not allow non-numeric input variables. But for a feature with categorical values, such as "Product Info 2", we cannot simply map its values into integers, because `scikit-learn` functions assume the order between integer values is relevant. So such features are mapped into several additional variables, called dummy variables. See `OneHotEncoder` and `get_dummies`. Encode all categorical variables into dummies using either of the two functions.

`get_dummies` has a "drop first" option. Why is this useful? (Hint: $\mathbf{X}$ should be full column rank.)

(d) Plot a `validation_curve` for `DecisionTreeClassifier` with varying values for the `max_depth` parameter. Determine the best choice for `max_depth` from your plot. Next, with `max_depth` set to its empirically optimal value, plot the `learning_curve` for the `DecisionTreeClassifier`. For this, and all remaining questions, compute each score using 5-fold cross validation.

(e) Repeat above but for `LogisticRegression`, in which vary the parameter `C` in the validation curve and use the optimal choice for `C` in the learning curve. Compare the learning curves for DecisionTreeClassifier and `LogisticRegression` and discuss which of the two has (i) more bias and (ii) more variance.

Optional problems, do not submit

6. Although the Prudential problem looks like a classification problem, the output variable is neither categorical not continuous. It is discrete, and the order between the values 1, ..., 8 is relevant. So linear regression can also be applied. Use `ElasticNet`—linear regression with a combination of L_1 and L_2 regularization. Apply `GridSearchCV` on ElasticNet to determine the best combination of `alpha` and `l1_ratio` parameters.

7. A simple improvement to linear regression for the Prudential problem

is to force all values to be one of the eight values, i.e., quantization. Implement quantization on "top of ElasticNet", using either sub-classing or pipelines with custom transforms.