Massachusetts Institute of Technology

6.867 Machine Learning, Fall 2006

Problem Set 3

Due Date: Monday, Oct 30, 1:05pm

You may submit your solutions in class or in the box.

1. We will examine here leave-one-out cross-validation as a model selection tool. Let $S_n = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ denote our training set and S_n^{-i} the corresponding set with the i^{th} training example and label removed. LOOCV is performed as follows: for each $(\mathbf{x}_i, y_i)$ in the training set, we train the classifier on the remaining $n-1$ points S_n^{-i} and test our prediction on the left-out pair $(\mathbf{x}_i, y_i)$. More formally, when using the squared loss, we define $error_{LOOCV}$ as

$$error_{LOOCV}(S_n) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}_{-i}(x_i))^2 \tag{1}$$

where $\hat{f}_{-i}$ is the estimator trained on S_n^{-i}.

(a) Let's start with a simpler strategy. We only leave out the first point, i.e., train with S_n^{-1}, and test on $(\mathbf{x}_1, y_1)$. The error is now

$$error_1(S_n) = (y_1 - \hat{f}_{-1}(\mathbf{x}_1))^2 \tag{2}$$

Assuming each training example and label is sampled independently from some underlying distribution $P(\mathbf{x}, y)$, show that

$$E\{error_1(S_n)\} = E\{(y - \hat{f}_{S_{n-1}}(\mathbf{x}))\}^2 \tag{3}$$

where the expectation on the left is over all random quantities and, on the right hand side, it is over both $(\mathbf{x}, y)$ (test example) as well as a dataset S_{n-1} of size $n-1$ sampled from the same distribution. In other words, on average, $error_1(S_n)$ gives us the test error!

(b) Now, using the above result, show that $error_{LOOCV}(S_n)$ also has this property, i.e.,

$$E\{error_{LOOCV}(S_n)\} = E\{(y - \hat{f}_{S_{n-1}}(\mathbf{x}))^2\} \tag{4}$$

(c) Parts a) and b) seem to indicate that both LOOCV and the single test set approximation are unbiased estimates of the test error based on $n-1$ training examples. Are the variances of $error_{LOOCV}$ and $error_1$ the same as well?

We now consider a situation where cross-validation can be ineffective as a model selection strategy. Suppose we have a classification task where we have binary-valued labels and binary-valued d-dimensional examples. In other words, $\mathbf{x}_i \in \{-1, 1\}^d$ and $y_i \in \{-1, 1\}$. There are d models, each making use of only one feature (coordinate) of $\mathbf{x}$. Model $\mathcal{M}_k$ corresponding to coordinate k can produce one of two possible estimators:

$$\mathcal{M}_k = \{f^k_{keep}, f^k_{flip}\} \quad (5)$$
$$f^k_{keep}(\mathbf{x}) = x_k \quad (6)$$
$$f^k_{flip}(\mathbf{x}) = -x_k \quad (7)$$

where x_k is the k-th feature/coordinate of $\mathbf{x}$.

Given any dataset S_n, the model $\mathcal{M}_k$ chooses the estimator (f^k_{flip} or f^k_{keep}) which results in the lowest training error, i.e., the final estimator $\hat{f}^k_{S_n}$ is selected on the basis of whether x_k or $-x_k$ better agrees with y.

Now, suppose that the data was generated is as follows: $Pr(x_k = 1) = 0.5$ for all $k = 1, \ldots, d$, i.e, the coordinates of $\mathbf{x}$ are sampled uniformly at random from $\{-1, 1\}^d$. The y values are generated from the $\mathbf{x}$ values (based on only one coordinate) in a probabilistic fashion, the details of which are not known.

Our goal is to use LOOCV to identify the best model $\mathcal{M}_k$ for classification.

(d) What is the probability that model $\mathcal{M}_r$ relying on an irrelevant coordinate r will produce an estimator with zero training error?
Hint: What value(s) must the vector $(x_{1r}, x_{2r}, \ldots, x_{nr})$take to ensure that no mistakes occur during training? Here x_{ir} is the r^{th} coordinate of the i^{th} example.

(e) For any $\mathcal{M}_r$ with training error ϵ (where $\epsilon \ll \frac{1}{2}$) show that the training error = $error_{LOOCV}$.
Hint: what can you say about the estimator $\hat{f}^k_{-i}$ produced in a LOOCV step?

(f) How many dimensions d do we need so that with probability 1/2 at least one model $\mathcal{M}_r$ based on an irrelevant coordinate would have $error_{LOOCV} < \epsilon$ (again, assume $\epsilon \ll \frac{1}{2}$)? An upper bound suffices.

2. We will explore here the use of marginal likelihood for feature selection with a simple "voting" classifier. Suppose we have $d-$dimensional binary input examples $\mathbf{x}$ where each coordinate $x_j \in \{-1, 1\}$. Our goal is to select a subset of features (coordinates of $\mathbf{x}$) so as to ensure that the classifier generalizes well. In particular, we are interested in how marginal likelihood might be able to guide us in this process. If $\mathcal{J}$ denotes the indexes of features we have chosen, then the discriminant function takes the form

$$f(\mathbf{x}; \theta, \mathcal{J}) = \frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} \theta_j x_j \quad (8)$$

We will assume for simplicity that the parameters are also binary so that $\theta_j \in \{-1, 1\}$. The product $\theta_j x_j$ in this case specifies the ± 1 label that the j^{th} coordinate is voting for. We can define a probability distribution over y based on the value of the discriminant function according to

$$P(y|\mathbf{x}, \theta, \mathcal{J}) = \frac{1}{2}\Big(1 + y f(\mathbf{x}; \theta, \mathcal{J})\Big) = \frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} (1 + y\theta_j x_j)/2 \quad (9)$$

In other words, the probability distribution over the labels is based on counting how many of the coordinate predictions agree with the label. Note that $(1 + y\theta_j x_j)/2 = 1$ if the j^{th} coordinate prediction agrees with y and zero otherwise.

We will try to use the marginal likelihood to select the appropriate coordinates, i.e., to find $\mathcal{J}$. Before we can evaluate the marginal likelihood we have to specify a prior distribution $P(\theta)$ over the binary parameters θ. If we have no reason to prefer one set of parameter values over another, we can just assume that all binary parameter vectors are equally likely:

$$P(\theta|\mathcal{J}) = \left(\frac{1}{2}\right)^{|\mathcal{J}|} \tag{10}$$

Now, given a training set $S_n = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ of examples and labels, we can in principle evaluate the marginal likelihood for any subset $\mathcal{J} \subseteq \{1, \ldots, d\}$:

$$P(S_n|\mathcal{J}) = \sum_{\theta \in \{-1,1\}^{|\mathcal{J}|}} P(\theta|\mathcal{J}) \prod_{t=1}^{n} P(y_t|\mathbf{x}_t, \theta, \mathcal{J}) \tag{11}$$

$$= \sum_{\theta \in \{-1,1\}^{|\mathcal{J}|}} \left(\frac{1}{2}\right)^{|\mathcal{J}|} \prod_{t=1}^{n} \left[\frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} (1 + y_t \theta_j x_{tj})/2\right] \tag{12}$$

where x_{tj} is the j^{th} coordinate of $\mathbf{x}_t$.

(a) Evaluate an expression for $P(S_n|\mathcal{J})$ when $\mathcal{J}$ is a singleton set, i.e., when $\mathcal{J} = \{l\}$ for some $l \in \{1, \ldots, d\}$.

(b) Do you see a problem? Can you propose how we should fix it?

(c) In directory hw3 you can find a MATLAB function `logmarlikel(X,y,idx)` that evaluates the logarithm of the above marginal likelihood (and includes a particular "fix" to the problem). The matrix argument `X` contains the input vectors as rows, `y` is a vector of corresponding labels, and `idx` specifies the columns of `X` we care about. Let's try to see that our score behaves reasonably. Load `Xrand.dat` and `yrand.dat`. These are randomly generated and should not contain any real relationship between $\mathbf{x}$ and y. Plot the log-marginal likelihoods corresponding to the following feature sets `(1)`, `(1:2)`, ..., `(1:10)`. What can you say about how the log-marginal likelihood behaves as a function of the size of the set? Is the behavior reasonable? Try using `logmarlikel(X,y,idx,0.4)` (assuming high label noise).

(d) It is hard to search over all possible subsets of features but we can do it sequentially. That is, we can find the best single feature first, then find another one to add that works best in combination with the first, and so on. When would we stop?

(e) Set `Xrand(:,2) = Yrand` so that the second column of `Xrand` now contains the labels to be predicted. Rerun part a) with this data. Explain the result.