Massachusetts Institute of Technology

6.867 Machine Learning, Fall 2006

Problem Set 4

Due Date: Thursday, Nov 9, 12:00 noon

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

1. We will explore here the use of the Bayesian Information Criterion (BIC) for model selection, specifically to select features for a Naive Bayes model. The input consists of d-dimensional binary feature vectors $\mathbf{x} = [x_1, \ldots, x_d]^T$ where $x_i \in \{-1, 1\}$ and binary class labels $y \in \{-1, 1\}$.

   A Naive Bayes model may be used, for example, to classify emails as spam ($y = 1$) or not-spam ($y = -1$). The body of each email can be represented as a bag of words (i.e., we ignore frequency, placement, and grammar etc.). We could restrict ourselves to a dictionary of d words $\{w_1, \ldots, w_d\}$. and coordinate x_i could serve as an indicator of word w_i: $x_i = 1$ if w_i is present in the email and $x_i = -1$ if it is absent.

   Recall that the Naive Bayes model assumes that the features are conditionally independent given the label so that

$$P(\mathbf{x}, y) = \left[\prod_{i=1}^{d} P(x_i|y)\right] P(y) \tag{1}$$

   We parameterize our model by introducing separate parameters for each component:

$$\begin{aligned} P(\mathbf{x}|y, \theta) &= \left[\prod_{i=1}^{d} P(x_i|y, \theta_i)\right] \quad \text{where} & (2)\\ P(x_i|y, \theta_i) &= \theta_{i|y}^{\frac{x_i+1}{2}} (1 - \theta_{i|y}^{\frac{1-x_i}{2}}) & (3)\\ & & (4) \end{aligned}$$

   Here $\theta = [\theta_1, \ldots, \theta_d]^T$ and $\theta_i = [\theta_{i|1}, \theta_{i|-1}]^T$ so that

$$\begin{aligned} \theta_{i|1} &= P(x_i = 1|y = 1)\\ \theta_{i|-1} &= P(x_i = 1|y = -1) \end{aligned}$$

   In class, we have already discussed how to compute the Maximum Likelihood estimates of these parameters. We will take a Bayesian approach, however, and introduce a prior probability over the parameters θ as:

$$P(\theta) = \prod_{i=1}^{d} \prod_{y \in \{1,-1\}} P(\theta_{i|y}) \tag{5}$$

$$P(\theta_{i|y}) = \frac{\Gamma(r^+ + r^- + 2)}{\Gamma(r^+ + 1)\Gamma(r^- + 1)} \cdot \theta_{i|y}^{r^+} (1 - \theta_{i|y})^{r^-} \tag{6}$$

where r^+ and r^- are *hyper-parameters*, common across all the $\theta_{i|y}$'s, and, for integer k, $\Gamma(k+1) = k!$. The hyper-parameters characterize our beliefs about the parameters prior to seeing any data. You may assume here that r^+ and r^- are non-negative integers.

(a) Eqn 5 specifies the prior $P(\theta_{i|y})$ as a Beta distribution. This choice of the prior is particularly convenient, as we will see now. Show that the posterior distribution $P(\theta|\mathcal{D})$, given n training examples $\{(\mathbf{x}_j, y_j) \mid j = 1, \ldots, n\}$, has the following form:

$$P(\theta|\mathcal{D}) \propto L(\mathcal{D};\theta)P(\theta) \quad = \quad \prod_{i=1}^{d} \prod_{y\in\{-1,1\}} \theta_{i|y}^{m^+_{i|y}} (1-\theta_{i|y})^{m^-_{i|y}}$$

where $L(\mathcal{D};\theta)$ is the likelihood function. What are $m^+_{i|y}$ and $m^-_{i|y}$? Your answer should use the $\hat{n}_y$ and $\hat{n}_{iy}(x_i, y)$ notation used in the lectures; e.g., $\hat{n}_{ky}(1,-1)$ is the number of examples where $y = -1$ and $x_k = 1$.

You have just shown that the Beta distribution is a conjugate prior to the Binomial distribution. In other words, if the prior probability $P(\theta)$ has the form of a Beta distribution, and the likelihood $P(\mathcal{D}|\theta)$ has the form of a Binomial distribution, then the posterior probability $P(\theta|\mathcal{D})$ will be a Beta distribution. When the class labels and features are not binary but take on k values, the same relationship holds between the Dirichlet and Multinomial distributions.

(b) Recall that, in the Bayesian scheme for model selection, our aim is to find the model that maximizes the *marginal likelihood*, $P(\mathcal{D}|\mathcal{F}_d)$, which is the normalization constant for the posterior:

$$P(\mathcal{D}|\mathcal{F}_d) \quad = \quad \int L(\mathcal{D};\theta)P(\theta)d\theta$$

where $\mathcal{F}_d$ denotes the model. Compute a closed-form expression for $P(\mathcal{D}|\mathcal{F}_d)$. (Hint: use the form of the normalization constant for the beta distribution).

(c) Let us now use our results to choose between two models $\mathcal{F}_1$ and $\mathcal{F}_2$. The only difference between them is in how they use feature i: $\mathcal{F}_2$ involves $P(x_i|y)$ term that connects the feature to the label y whereas $\mathcal{F}_1$ only has $P(x_i)$, assuming that feature x_i is independent of the label. In $\mathcal{F}_2$ the distribution of x_i's is parameterized by two parameters $[\theta_{i|1}, \theta_{i|-1}]^T$ as described before. In contrast, $\mathcal{F}_1$ only requires a single parameter θ'_i,

$$P(x_i|\theta'_i) \quad = \quad \theta_i'^{\frac{x_i+1}{2}}(1-\theta'_i)^{\frac{1-x_i}{2}} \tag{7}$$

The prior probability over θ'_i is the same as Eqn 6:

$$P(\theta'_i) \quad \propto \quad \theta_i'^{r^+}(1-\theta'_i)^{r^-} \tag{8}$$

The expressions of the marginal likelihood from the two models will differ only in terms of feature i. To compare the two models we can ignore all the other terms. From your results in part (b), extract from $P(\mathcal{D}|\mathcal{F}_2)$ the factors involving feature i. Use a similar analysis to evaluate the terms involving feature i in $P(\mathcal{D}|\mathcal{F}_1)$. Using these, evaluate the "decision rule", $\log[P(\mathcal{D}|\mathcal{F}_2)/P(\mathcal{D}|\mathcal{F}_1)] > 0$ to include feature x_i.

(d) **Optional (Extra Credit):** For many models the marginal likelihood is difficult to compute exactly and we have to an asymptotic approximation:

$$BIC(\mathcal{D}, \mathcal{F}_r) \quad = \quad \log L(\mathcal{D}; \hat{\theta}_{ML}) - \frac{r}{2} \log n$$

We will now see (roughly speaking) how to relate BIC and $P(\mathcal{D}|\mathcal{F})$ in the general case. Our intuition is as follows: as the number of examples n increases the posterior distribution $P(\theta|\mathcal{D})$ becomes more and more peaked around the maximum likelihood estimate $\hat{\theta}_{ML}$. In the limit $n \to \infty$, the posterior distribution becomes a point estimate at $\hat{\theta}_{ML}$. We then approximate the integral by evaluating it only in the neighbourhood of $\hat{\theta}_{ML}$. We start by writing:

$$P(\mathcal{D}|\mathcal{F}_r) \quad = \quad \int \exp(\log L(\mathcal{D}; \theta)) P(\theta) d\theta$$

and perform a Taylor series expansion of $\log L(\mathcal{D}; \theta)$ around $\hat{\theta}_{ML}$:

$$\begin{aligned}
\log L(\mathcal{D}; \theta) &= \log L(\mathcal{D}; \hat{\theta}_{ML}) + (\theta - \hat{\theta}_{ML})^T A_1 + \frac{1}{2}(\theta - \hat{\theta}_{ML})^T A_2 (\theta - \hat{\theta}_{ML}), \text{ where} \\
A_1 &= \left. \frac{\partial \log L(\mathcal{D}; \theta)}{\partial \theta} \right|_{\hat{\theta}_{ML}} \\
A_2 &= \left. \frac{\partial^2 \log L(\mathcal{D}; \theta)}{\partial \theta \partial \theta^T} \right|_{\hat{\theta}_{ML}}
\end{aligned}$$

The matrix A_2 has interesting properties (it's called the Fisher Information Matrix), but its most relevant property here is that the determinant $|A_2| \approx nC(r)$ where $C(r)$ does not depend on the number of examples n.

We have performed a second-order Taylor series expansion, i.e. up to the second derivative. What would the problem be if we had performed only a first-order expansion?

(e) **Optional (Extra Credit):** Now, assume that the prior is uninformative, i.e $P(\theta) = 1$ (strictly speaking, it should be $1/(\int d\theta)$, but the constant factor will not matter). Show that with the second order approximation

$$\int L(\mathcal{D}; \theta) P(\theta) d\theta \quad \approx \quad L(\mathcal{D}; \hat{\theta}_{ML}) \left(\frac{2\pi}{n}\right)^{d/2} C_1(r)$$

where $C_1(r)$ consists of terms that do not depend on the number of examples n.

(f) **Optional (Extra Credit):** Using your result from part (g), argue that

$$\lim_{n \to \infty} \log P(\mathcal{D}|\mathcal{F}_r) \quad = \quad BIC(\mathcal{D}, \mathcal{F}_r)$$

This is the motivation behind the use of the BIC score for model selection.

2. Kenny and Cartman are playing a game; in this game, Cartman draws samples from a normal distribution and gives them to Kenny, who must determine what the parameters (i.e., the mean and variance) of the distribution are. Kenny knows that he can simply report the sample mean and sample variance,

as they are the maximum-likelihood estimators for the mean and variance. However, it turns out that Cartman is cheating. Instead of keeping the parameters fixed, he occassionally changes them without telling Kenny.

Kenny is, however, very cunning, and catches on. He decides to use BIC to determine when Cartman is changing the parameters.

(a) Given data $X_1, X_2, ..., X_N$, write a formula for the BIC for the hypothesis that there are m change-points t_1, t_2, ..., t_m, with

$$X_i \sim N(\mu_k, \Sigma_k) \qquad \text{when } t_{k-1} \leq i < t_k, \tag{9}$$

where $t_0 = 0$ and $t_{m+1} = N$ for convenience.

(b) How can Kenny use this to determine the number of change-points and where they are? Write a MATLAB program to execute this procedure assuming that there is at most one change-point and plot the BIC as a function of the hypothesized change-point for the dataset `hw4p3b`. Your algorithm should run in linear time in the number of samples!

(c) Kenny realizes that this procedure is broadly applicable. He finds an audio file of three students saying hello and, by assuming that students generate speech by drawing cepstra from multivariate-normal distributions, whose parameters depend on things like the student's vocal chords and their position relative to the microphone and so on, he discovers that he can use BIC to segment the audio file to find a speaker change! What's more, using a sliding-window algorithm that his TA provided (`multisplit.m`), he finds that he can find multiple change-points. Try it out! (Use the data in `hw4p3d`.) How many speaker changes are there and when do they occur? Now try `hw4p3e`. What are the speaker changes now? Do they make sense? If not, what went wrong?

3. AdaBoost is a simple algorithm for estimating ensembles

$$h_m(\mathbf{x}) = \sum_{j=1}^{m} \alpha_j h(\mathbf{x}; \theta_j) \tag{10}$$

where $\alpha_j \geq 0$ (not necessarily summing to one) and we have so far assumed that each $h(\mathbf{x}; \theta_j)$ is a decision stump. The basic boosting algorithm can be easily extended, however. For example, we can replace the decision stumps with different base classifiers and/or we can use a different loss function to train the ensemble. The exponential loss used in AdaBoost is not very forgiving about outliers and so we will try to use another loss function, along with different base learners.

At stage m of the algorithm, we we will try to minimize

$$J(\alpha_m, \theta_m) = \sum_{t=1}^{n} \text{Loss}\Big(y_t h_m(\mathbf{x}_t)\Big) = \sum_{t=1}^{n} \text{Loss}\Big(y_t h_{m-1}(\mathbf{x}_t) + y_t \alpha_m h(\mathbf{x}_t; \theta_m)\Big) \tag{11}$$

where we assume that $h_{m-1}(\mathbf{x})$ is fixed from $m-1$ previous rounds and view J as a function of parameters α_m and θ_m associated with the new base classifier. The margin loss Loss$(\cdot)$ could be any function that is convex and decreasing in its argument (smaller loss for predictions with larger voting margin).

Let's elaborate on the revised algorithm a bit. We can initialize $h_0(\mathbf{x}) = 0$ for the ensemble with no base learners. Then, after $m-1$ rounds, we perform two distinct steps to optimize θ_m and α_m, respectively.

First, we find parameters $\hat{\theta}_m$ that minimize

$$\frac{d}{d\alpha_m} J(\alpha_m, \theta_m)\bigg|_{\alpha_m=0} = \sum_{t=1}^{n} \overbrace{\mathrm{dL}\Big(y_t h_{m-1}(\mathbf{x}_t)\Big)}^{-W_{m-1}(t)} y_t h(\mathbf{x}_t; \theta_m) = -\sum_{t=1}^{n} W_{m-1}(t)\, y_t h(\mathbf{x}_t; \theta_m) \tag{12}$$

where $\mathrm{dL}(z) = d/dz\,\mathrm{Loss}(z)$ is always negative because the loss is decreasing. The form of the optimization problem for θ_m is exactly as in AdaBoost except that the weights will be different due to the different margin loss. We can also normalize the weights in the algorithm since the normalization would not affect the resulting $\hat{\theta}_m$.

Once we have $\hat{\theta}_m$, we can find $\hat{\alpha}_m$ that minimizes

$$J(\alpha_m, \hat{\theta}_m) = \sum_{t=1}^{n} \mathrm{Loss}\left(y_t h_{m-1}(\mathbf{x}_t) + y_t \alpha_m h(\mathbf{x}; \hat{\theta}_m)\right) \tag{13}$$

This is easy to solve since $\mathrm{Loss}(\cdot)$ is convex and we only have one parameter to optimize.

We can now write the general boosting algorithm more succinctly. After initializing $\tilde{W}_0(t) = 1/n$, each boosting iteration consists of the following three steps:

(Step 1) Find $\hat{\theta}_m$ that (approximately) minimizes the weighted error ϵ_m or $2\epsilon_m - 1$ given by

$$-\sum_{t=1}^{n} \tilde{W}_{m-1}(t)\, y_t h(\mathbf{x}_t; \theta_m) \tag{14}$$

(Step 2) Find $\hat{\alpha}_m$ that minimizes

$$J(\alpha_m, \hat{\theta}_m) = \sum_{t=1}^{n} \mathrm{Loss}\left(y_t h_{m-1}(\mathbf{x}_t) + y_t \alpha_m h(\mathbf{x}; \hat{\theta}_m)\right) \tag{15}$$

(Step 3) Reweight the examples

$$\tilde{W}_m(t) = -c_m\, \mathrm{dL}\left(y_t h_{m-1}(\mathbf{x}_t) + y_t \hat{\alpha}_m h(\mathbf{x}; \hat{\theta}_m\right) \tag{16}$$

where c_m normalizes the new weights so they sum to one.

Now that we have a boosting algorithm for any loss function, we can select a particular one. Specifically, we will consider the logistic loss:

$$\mathrm{Loss}(z) = \log\left(1 + \exp(-z)\right) \tag{17}$$

(a) Show that the unnormalized weights $W_m(t)$ from the logistic loss are bounded by 1. What can you say about the resulting normalized weights for examples that are clearly misclassified in comparison to those that are just slightly misclassified by the current ensemble?

(b) Suppose the training set is linearly separable and we would use a linear support vector machine (no slack penalties) as a base classifier. In the first boosting iteration, what would the resulting $\hat{\alpha}_1$ be?

(c) Let's consider a bit simpler base learners but still those that can be combined into strong predictors. So, we define our base learners as radial basis functions centered at particular training points, i.e.,

$$h(\mathbf{x}; \theta) = y_t \exp(-\beta \|\mathbf{x} - \mathbf{x}_t\|^2) \tag{18}$$

where the parameter θ simply identifies the training point t. These base learners return real valued predictions $h(\mathbf{x}; t) \in [-1, 1]$. Provide an argument for why an ensemble with n such base learners can in principle classify a set of n points in all possible ways.

(d) It seems that the boosting algorithm with such base learners might seriously overfit. Let's see if this is true in practice. We have provided you with data and all the necessary matlab code to run the boosting algorithm with the radial basis learners. The files are available in hw4/prob3. The function you need to run is `call_boosting.m`. Does it overfit?

(e) To understand this a bit better let's plot the voting margin errors as a function of the number of boosting iterations. Recall that a margin error at level ρ occurs whenever $y_t \tilde{h}_m(\mathbf{x}_t) - \rho \leq 0$ where α_j's in the ensemble $\tilde{h}_m(\mathbf{x}_t)$ are normalized to sum to one. Plot the margin errors at levels $\rho = 0.1$ and $\rho = 0.5$. You should be able to do this with only a minor modification to the code we have provided. Can you use these plots to explain your observations in part d)?