SMLDM HT 2014 - MSc Problem Sheet 4

1. In this question we will take a Bayesian approach to learning decision trees. Assume that we have a binary classification problem, with classes $\{0, 1\}$ and we have n data items $(x_i, y_i)_{i=1}^n$.

   Recall the greedy tree growing heuristics for decision trees: we start with the root, for each leaf node of the tree we find an optimal feature j and split point v, according to some criteria, to split the node, and recurse on both sides.

   (a) Consider a decision tree T where the data space has been split into R regions $\mathcal{R}_1, \ldots, \mathcal{R}_R$. Each region $\mathcal{R}_m$ corresponds to a leaf in T and is a rectangle, with side $[a_j, b_j]$ in the jth dimension, $j = 1, \ldots, p$, and has a parameter β_m which gives the probability of class 1 for data vectors in that region.

   What is a conjugate prior for the parameters β? Calculate the marginal probability $p(\mathbf{Y}|\mathbf{X}, T)$ of the responses given the data vectors and tree T, marginalizing out β.

   Answer: The beta distibution $\mathrm{Beta}(\alpha_0, \alpha_1)$ is conjugate to the Bernoulli likelihood parametrized by β. Let $c_{mk} = \sum_{i=1}^n 1[x_i \in \mathcal{R}_m] 1[y_i = k]$ for $m = 1, 2, \ldots, R$ and $k = 0, 1$, that is, c_{m0}, c_{m1} denote the number of 0s and 1s in leaf node m respectively. Since the training data are i.i.d., each training data point contributes to only the likelihood of the leaf node where it belongs. Hence, the likelihood factorizes over the leaf nodes:

$$p(\mathbf{Y}|\mathbf{X}, T, \beta) = \prod_{m=1}^{R} \beta^{c_{m1}} (1-\beta)^{c_{m0}}.$$

   The marginal likelihood is given by

$$\begin{aligned} p(\mathbf{Y}|\mathbf{X}, T) &= \prod_{m=1}^{R} \frac{1}{\mathcal{B}(\alpha_0, \alpha_1)} \int_\beta \beta^{c_{m1}+\alpha_1-1} (1-\beta)^{c_{m0}+\alpha_0-1} d\beta, \\ &= \prod_{m=1}^{R} \frac{\mathcal{B}(c_{m0}+\alpha_0, c_{m1}+\alpha_1)}{\mathcal{B}(\alpha_0, \alpha_1)}, \end{aligned}$$

   where $\mathcal{B}(\alpha_0, \alpha_1) = \frac{\Gamma(\alpha_0)\Gamma(\alpha_1)}{\Gamma(\alpha_0+\alpha_1)}$ is the beta function.

   (b) Consider a greedy model selection procedure for determining the structure of T:

   i. We start with a trivial tree with a single node.

   ii. At each iteration we consider expanding a leaf node m of the tree by creating a split at feature j, value v. This produces a tree T' with two more nodes than T, both children of node m.

   iii. We compute the marginal probability of $\mathbf{Y}$ under T', for each j and v, and find the split producing the highest marginal probability.

   iv. If the marginal probability of the resulting T' is larger than T, we split the node, otherwise we consider expanding other nodes.

   v. We stop once all leaf nodes of the current tree T have been considered for expansion, but all lead to trees T' with lower marginal probability than T.

   Calculate the marginal probability $p(\mathbf{Y}|\mathbf{X}, T')$ of the responses given the data vectors under tree T'.

Answer: Let T denote the current tree and T' denote the new tree where node m has been split into left and right child nodes, denoted by m_l and m_r respectively. Let $c_{m_l 0}, c_{m_l 1}$ denote the counts for child m_l (define counts similarly for m_r). Let $\ell(c_{m0}, c_{m1}) = \frac{\mathcal{B}(c_{m0}+\alpha_0, c_{m1}+\alpha_1)}{\mathcal{B}(\alpha_0,\alpha_1)}$ denote the likelihood for leaf m. The new marginal likelihood is given by

$$p(\mathbf{Y}|\mathbf{X}, T') = \ell(c_{m_l 0}, c_{m_l 1})\ell(c_{m_r 0}, c_{m_r 1}) \prod_{m'=1, m' \neq m}^{R} \ell(c_{m'0}, c_{m'1}),$$

(c) Explain how the ratio of marginal probabilities under T' and T (the so-called **Bayes factor**) simplifies to a function which depends only on the data items under region $\mathcal{R}_m$.

Answer: From the definitions of $p(\mathbf{Y}|\mathbf{X}, T')$ and $p(\mathbf{Y}|\mathbf{X}, T)$, it is straightforward to see that

$$\begin{aligned}\frac{p(\mathbf{Y}|\mathbf{X}, T')}{p(\mathbf{Y}|\mathbf{X}, T)} &= \frac{\ell(c_{m_l 0}, c_{m_l 1})\ell(c_{m_r 0}, c_{m_r 1}) \prod_{m'=1, m' \neq m}^{R} \ell(c_{m'0}, c_{m'1})}{\prod_{m'=1}^{R} \ell(c_{m'0}, c_{m'1})} \\ &= \frac{\ell(c_{m_l 0}, c_{m_l 1})\ell(c_{m_r 0}, c_{m_r 1})}{\ell(c_{m0}, c_{m1})},\end{aligned}$$

which clearly depends only on the data points in region $\mathcal{R}_m$.

(d) For each j, explain why the marginal probability under T' is a piecewise constant function of v.

Answer: Let v denote the split points for j. The marginal probability under T' depends on j, v only through the counts $c_{m_l 0}, c_{m_l 1}, c_{m_r 0}, c_{m_r 1}$ of the left and right children of the split j, v. Since the counts $c_{m_l 0}, c_{m_l 1}, c_{m_r 0}, c_{m_r 1}$ change only at the unique values of x_j within $\mathcal{R}_m$, the marginal probability under T' is a piecewise constant function of v.

(e) Describe an algorithm that can determine the optimal split of node m, with computational cost $p \times N_m$, where N_m is the number of data items in region m, and p the number of features.

Answer: For each dimension, we first sort the N_m data points along that dimension. The counts change only at unique values of x and sorting ensures that we know the range of valid split intervals. There are at most $N_m - 1$ valid split intervals (if all x values are unique). With a simple linear scan through the sorted data with $\mathcal{O}(N_m)$ complexity, we can update the counts for all valid split intervals along this dimension. Repeating this for all p dimensions, we can compute the quality of all valid splits and determine the optimal split in $\mathcal{O}(pN_m)$ time. Note that the sort operation takes $\mathcal{O}(pN_m \log N_m)$, but the $\log N_m$ factor is usually negligible compared to the cost of evaluating the quality of each split.

2. Recall the definition of a 1 hidden layer neural network for binary classification in the lectures. The objective function is:

$$J = -\sum_{i=1}^{n} y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) + \frac{1}{2} \sum_{jk} C|W_{jk}^h|^2 + \frac{1}{2} \sum_{k} C|W_k^o|^2$$

and the network definition is:

$$\hat{y}_i = s\left(b^o + \sum_{k=1}^{m} W_k^o h_{ik}\right) \qquad h_{ik} = s\left(b_k^h + \sum_{j=1}^{p} W_{jk}^h x_{ij}\right)$$

(a) Verify that the derivatives needed for gradient descent are:

$$\frac{dJ}{dW_k^o} = CW_k^o + \sum_{i=1}^{n}(\hat{y}_i - y_i)h_{ik}$$
$$\frac{dJ}{dW_{jk}^h} = CW_{jk}^h + \sum_{i=1}^{n}(\hat{y}_i - y_i)W_k^o h_{ik}(1 - h_{ik})x_{ij}$$

Answer: The first terms are the derivatives of the regularization terms. The second terms are derivatives of the empirical risk terms, which we will use the chain rule.

$$\begin{aligned}
\frac{dJ}{dW_k^o} &= CW_k^o + \sum_{i=1}^{n}\frac{dJ}{d\hat{y}_i}\frac{d\hat{y}_i}{dW_k^o} \\
&= CW_k^o + \sum_{i=1}^{n}\left(-\frac{y_i}{\hat{y}_i} + \frac{1-y_i}{1-\hat{y}_i}\right)(\hat{y}_i(1-\hat{y}_i))\,h_{ik} \\
&= CW_k^o + \sum_{i=1}^{n}(-y_i(1-\hat{y}_i) + (1-y_i)\hat{y}_i)h_{ik} \\
&= CW_k^o + \sum_{i=1}^{n}(\hat{y}_i - y_i)h_{ik} \\
\frac{dJ}{dW_{jk}^h} &= CW_{jk}^h + \sum_{i=1}^{n}\frac{dJ}{d\hat{y}_i}\frac{d\hat{y}_i}{dh_{ik}}\frac{dh_{ik}}{dW_{jk}^h} \\
&= CW_{jk}^h + \sum_{i=1}^{n}\left(-\frac{y_i}{\hat{y}_i} + \frac{1-y_i}{1-\hat{y}_i}\right)(\hat{y}_i(1-\hat{y}_i)W_k^o)\,(h_{ik}(1-h_{ik})x_{ij}) \\
&= CW_{jk}^h + \sum_{i=1}^{n}(\hat{y}_i - y_i)W_k^o h_{ik}(1-h_{ik})x_{ij}
\end{aligned}$$

(b) Suppose instead that you have an L layer neural network for binary classification, with each hidden layer having m neurons with logistic nonlinearity. Define carefully the network, giving the parameterization of each layer, and derive the backpropagation algorithm to compute the derivatives of the objective with respect to the parameters. You may ignore bias terms for simplicity.

Answer: The network is defined as follows. For simplicity, let $h_i^0 = x_i$ be the input vector, and h_i^ℓ be a vector which denotes the activations of the ℓth hidden layer. Finally let $h_i^{L+1} = \hat{y}_i$ be the predicted probability of class 1. We have:

$$h_i^{\ell+1} = s(W^{\ell+1}h_i^\ell)$$

where the parameters are $W^{\ell+1}$, a matrix of weights (of appropriate sizes) for each ℓ. The objective is:

$$J = -\sum_{i=1}^{n} y_i \log h_i^{L+1} + (1-y_i)\log(1-h_i^{L+1})$$

We ignore the regularization term as well as it is straightforward to deal with.

We will compute the derivatives with respect to the unit activations recursively, starting from layer $L+1$ and working backwards towards layer 0:

$$\frac{dJ}{dh_i^{L+1}} = -\frac{y_i}{h_i^{L+1}} + \frac{1-y_i}{1-h_i^{L+1}}$$

$$\frac{dJ}{dh_{ik}^{\ell}} = \sum_j \frac{dJ}{dh_{ij}^{\ell+1}} \frac{dh_{ik}^{\ell+1}}{dh_i^{\ell}}$$

$$= \sum_j \frac{dJ}{dh_i^{\ell+1}} h_{ij}^{\ell+1}(1-h_{ij}^{\ell+1}) W_{jk}^{\ell+1}$$

Finally, we can compute the derivatives with respect to the parameters:

$$\frac{dJ}{dW_{jk}^{\ell}} = \sum_{i=1}^{n} \frac{dJ}{dh_j^{\ell}} \frac{dh_j^{\ell}}{dW_{jk}^{\ell}}$$

$$= \sum_{i=1}^{n} \frac{dJ}{dh_j^{\ell}} h_j^{\ell}(1-h_j^{\ell}) h_k^{\ell-1}$$

3. A **mixture of experts** is a type model in which a number of experts "compete" to predict a label.

   Consider a regression problem with dataset $(x_i, y_i)_{i=1}^n$ and $y_i \in \mathbb{R}$. We have E experts, each being a parametrized function $f_j(x;\theta_j)$, for $j = 1, \ldots, E$. For example each expert could be a neural network. Each expert $f_j(x;\theta_j)$ tries to predict the response y corresponding to data vector x.

   (a) A simple mixture of experts model uses as it's objective function

   $$J(\pi, \sigma^2, (\theta_j)_{j=1}^E) = \sum_{i=1}^{n} \log \sum_{j=1}^{E} \pi_j e^{-\frac{1}{2\sigma^2}\|f_j(x_i;\theta_j)-y_i)\|^2}$$

   where $\pi = (\pi_1, \ldots, \pi_E)$ are mixing proportions and σ^2 is a parameter.

   Relate the objective function to the log likelihood of a mixture model where each component is a conditional distribution of Y given $X = x$.

   Answer: This is simply a mixture model, with the jth component being the conditional normal:

   $$p(y_i|x_i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}\|f_j(x_i;\theta_j)-y_i\|^2}$$

   and mixing proportions π_j. The objective is (up to the $2\pi\sigma^2$ constant) the log likelihood.

   (b) Differentiate the objective function with respect to θ_j, interpreting the computation of the derivative as a generalized EM algorithm, where in the E step the posterior distribution is computed, and in the M step gradient descent is used to update the expert parameters θ_j.

   Answer: The derivative wrt θ_j is:

   $$\frac{dJ}{d\theta_j} = \sum_{i=1}^{n} \frac{\pi_j e^{-\frac{1}{2\sigma^2}\|f_j(x_i;\theta_j)-y_i)\|^2}}{\sum_{j=1}^{E} \pi_j e^{-\frac{1}{2\sigma^2}\|f_j(x_i;\theta_j)-y_i)\|^2}} \left(\frac{1}{\sigma^2}(y_i - f_j(x_i;\theta_j)) \nabla_{\theta_j} f_j(x_i;\theta_j) \right)$$

   The fraction is the responsibility of expert j for data item i, and the term in parentheses is the derivative of the log probability of y_i under expert j, wrt the parameter θ_j. This is the generalized M step.

(c) A mixture of experts allows each expert to specialize in predicting the response in a certain part of the data space, with the overall model having better predictions than any one of the experts.

However to encourage this specialization, it is useful also for the mixing proportions to depend on the data vectors x, i.e. to model $\pi_j(x;\phi)$ as a function of x with parameters ϕ. The idea is that this **gating network** controls where each expert specializes. To ensure $\sum_{j=1}^{E} \pi_j(x;\phi) = 1$, we can use the softmax nonlinearity:

$$\pi_j(x;\phi) = \frac{\exp(g_j(x;\phi_j))}{\sum_{\ell=1}^{E} \exp(g_\ell(x;\phi_\ell))}$$

where $g_j(x;\phi_j)$ are parameterized functions for the gating network.

The previous generalized EM algorithm extends to this scenario easily. Describe the latent variables you will need to introduce into the system, the free energy lower bound on the log likelihood, and derive the E step and generalized M step for ϕ_j from the free energy.

Answer: To be explicit about the EM link, we introduce a latent variable z_i to indicate which expert is responsible for predicting y_i. The joint probability of y_i, z_i given x_i is

$$\prod_{j=1}^{E} \left(\pi_j(x_i;\phi) \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}\|y_i - f_j(x_i;\theta_j)\|^2} \right)^{\mathbf{1}(z_i=j)}$$

The free energy is

$$\mathcal{F}(\theta,\phi,q) = \mathbb{E}\left[\sum_{i=1}^{n} \sum_{j=1}^{E} \mathbf{1}(z_i = j) \left(\log \pi_j(x_i;\phi) - \frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\|y_i - f_j(x_i;\theta_j)\|^2 \right) - \log q(\mathbf{z}) \right]$$

In the E step, the posterior is:

$$q(z_i = j) \propto \pi_j(x_i;\phi) e^{-\frac{1}{2\sigma^2}\|y_i - f_j(x_i;\theta_j)\|^2}$$

And in the generalized M step, the derivative wrt ϕ_j is:

$$\sum_{i=1}^{n} (q(z_i = j) - \pi_j(x_i;\phi)) \nabla_{\phi_j} g_j(x_i;\phi_j)$$