Course 02457 Non-Linear Signal Processing, Exercise 6

This exercise is based on C. M. Bishop: *Pattern Recognition and Machine Learning*, Chapter 5. Print and comment on the figures produced by the software as outlined below at the **Checkpoints**. You are going to use the 'Neural Classification' Matlab toolbox programmed at Cognitive Systems at DTU Informatics.

Pattern recognition with neural networks

The neural network for classification is designed to approximate posterior class probabilities. This is achieved using the transformation from the neural network outputs ϕ to the probabilities y by the normalized exponential or so-called *softmax* function:

$$y_k = \frac{\exp(\phi_k)}{\sum_j^c \exp(\phi_j)} \; . \tag{1}$$

This function has a built-in redundancy: If there are c classes and we use c outputs we can calculate the output of the c'th unit from the sum of the others. A way to overcome this redundancy is via a *modified softmax* which only has $c-1$ outputs from the neural network (before the softmax). The c'th output is then calculated from the others:

$$y_k = \frac{\exp(\phi_k)}{\sum_j^{c-1} \exp(\phi_j) + 1} \; , \qquad k = 1, 2, \ldots, c-1 \tag{2}$$

$$y_c = 1 - \sum_k^{c-1} y_k \; . \tag{3}$$

Pima indian data set

This is a data set where the task is to classify a population of women according to the risk of diabetes (binary classification). There are 7 input variables, 200 training examples and 332 test examples. 68 (34%) in the training set and 109 (32.82%) in the test set have been diagnosed with diabetes. In Brian Ripley's textbook *Pattern Recognition and Neural Networks* he states that his best method obtains about 20% misclassification on this data set so this is what we can use for reference. The input variables are:

1. Number of pregnancies
2. Plasma glucose concentration
3. Diastolic blood presure
4. Triceps skin fold thickness
5. Body mass index (weight/height^2)
6. Diabetes pedigree function
7. Age

The target output is 1 for examples having diabetes ("positive" diagnosis), and 0 for healthy subjects ("negative" diagnosis). The variables are not directly fed into the neural network. Instead the mean and standard deviation are calculated on the training set and these values are then used to normalize both the training and the test set, by first subtracting the mean and the dividing by the standard deviation. This procedure is called standardization, see Bishop page 567.

It is not possible to view all 7 dimension in one plot. A way to plot the data is to select two variables and plot them against each other. Another way is to use principal component analysis and plot the data projected one principal component against another.

Checkpoint 6.1

Use the program `main6a.m` to optimize a neural network and plot the decision boundary and conditional probabilities. The heavy blue line is for locating the contour line corresponding to the equal probability of the two classes.

Run the network with different weight decays and comment on the difference in the decision surfaces. Create a flow chart of `main6a.m` and its functions.

Entropic cost function

One could use the squared error function to optimize the neural network for classification:

$$E_{\text{square}} = 1/2 \sum_{n}^{N} \sum_{k}^{c} (y_k^n - t_k^n)^2 \tag{4}$$

The square error cost function is linked to the 'additive Normal noise' hypothesis. However, as $\mathbf{y}$ is supposed to model a probability and $\mathbf{t} = (0, 1, 0, .., 0)$ is a position coded target vector, the Gaussian error is inappropriate. Instead we can write the conditional likelihood function:

$$p(\mathbf{t}^n|\mathbf{x}^n) = \prod_{k}^{c} [y_k^n]^{t_k^n} \qquad \text{(Maximize)} \tag{5}$$

This will give the so-called *cross-entropy* cost function appropriate for classification. The negative log-likelihood of this is for an entire data set of independent samples:

$$E_{\text{entropy}} = -\sum_{n}^{N} \sum_{k}^{c} t_k^n \ln y_k^n \qquad \text{(Minimize)} \tag{6}$$

Checkpoint 6.2

The program `main6b.m` calculates the cost function value associated with one example for a two-class problem. It sweeps the output of the neural network ϕ_1 from -10 to 10.

What was the target t_1? How much do the two cost functions penalize a "wrong" decision.

Posterior probabilities

In the neural network we model the posterior class probabilities $p(\mathcal{C}_k|\mathbf{x})$ directly. However we can also start from the class conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ and then use Bayes theorem to convert this into posterior class probabilities:

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)\, p(\mathcal{C}_k)}{p(\mathbf{x})} \,. \tag{7}$$

This gives us a way for generating ground truth data to test the ability of the neural network to learn posterior class probabilities.

Checkpoint 6.3

Use the program `main6c.m` to plot posterior probabilities. The data is very simple with one dimensional $\mathbf{x}$ and two classes each Gaussian distributed with $\sigma^2_{\mathcal{A}} = \sigma^2_{\mathcal{B}} = 1$, $\mu_{\mathcal{A}} = -2$ and $\mu_{\mathcal{B}} = 2$, thus $p(\mathbf{x}|\mathcal{C}_{\mathcal{A}}) \sim \mathcal{N}(-2, 1)$ and $p(\mathbf{x}|\mathcal{C}_{\mathcal{B}}) \sim \mathcal{N}(2, 1)$. There are 100 samples in $\mathcal{A}$ and 12 samples in $\mathcal{B}$, initially, change this to larger values (e.g. 500 and 60, to keep the prior probabilities the same) and observe how the posterior approaches the true posterior.

Pruning

The program `main6d.m` will train and prune connections to optimize a neural network on the Pima indian data-set. The network is a multi-layer perceptron with softmax normalization.

We base pruning on the so-called 'optimal brain damage' (OBD) approach. The saliency of each weight is estimated using a second order Taylor expansion of the cost function around its minimum. This estimates the saliency (cost of removing the weight) of weight number q as $S_q = \frac{1}{2}\frac{\partial^2 E}{\partial w_q^2} w_q^2$.

The pruning procedure trains the initial configuration which fully connected between inputs and hidden layer nodes and between hidden layer and outputs nodes. The saliencies of all weigths are computed and the least salient weight is pruned (put permanently to zero). The pruned network is retrained and the procedure repeated until a minimal configuration is reached. During pruning the test error is measured and after pruning the best network in test is selected.

Checkpoint 6.4

Derive the saliency expression as the approximate cost of removing a single weight while keeping the remaining weights fixed at their 'optimal' values.

The best net is selected as the net with the lowest cost function value on the test set.

Note that not all inputs are used, is the number of pregnancies important for the diagnosis?

The target output is 1 for positive diagnosis (diabetes), and 0 for negative diagnosis (healthy).

What does the neural network say about the relation between age and diabetes (positive/negative correlation)?

Challenge I (not part of curriculum)

Modify the set-up of Checkpoint 6.3 to have different variances for the two Gaussians: $\sigma^2_{\mathcal{A}} = 2$ and $\sigma^2_{\mathcal{B}} = 1$. Calculate and plot the posterior probability in this case (hint: quadratic discriminant) , redo the experiment with 100+12 and 500+60 examples and comment on how the neural network approximates the posterior.

Challenge II (not part of curriculum)

Drop-out `http://arxiv.org/abs/1207.0580` is a new regularisation technique for neural networks. Modify the neural network code so that it performs drop-out regularization and apply it with standard settings (50% drop-out) to the Pima Indian dataset. Do you see any differences in the decision boundaries to the ones found in Checkpoint 6.1 and what is the test performance compared to the other schemes we have considered. Can you explain the mechanism behind drop-out and why this in some cases will lead to better performance?