Artificial neural networks

Assignment 2: Bayesian learning in neural networks

Lood, Cédric
Master of Bioinformatics

May 5, 2016

1 Context

Bayesian statistics offer a robust framework in which it is possible to build and assess models, and perform inference.

In this exercise, we were asked to explore bayesian learning applied to neural networks. Using the Bayes rule in the context of learning has several desirable properties related to the fact that one is not using point estimates in the search space of solutions (for example the weight space) but tries to obtain a distribution over the weight space. Hence, quantification of certainty of predictions is possible without the use of validation set or resampling.

2 BayesNN Demo

The figure 1, produced with the script [1] displays an example of classification of 4 labelled points. Starting from a gaussian, uninformative prior for the weights distribution. The algorithm computes first the posterior after having seen 2 datapoints $(-5,-5)$ and $(5,5)$. The posterior distribution takes then the shape of a sigmoidal ridge, with half the space with weights for w_1 and w_2 close to 0. Those weights would create a classifier whose slope is positive, hence failing to classify the 2 points. After seeing the other points (which are impossible to classify linearly), we can see the posterior peaking a bit more.

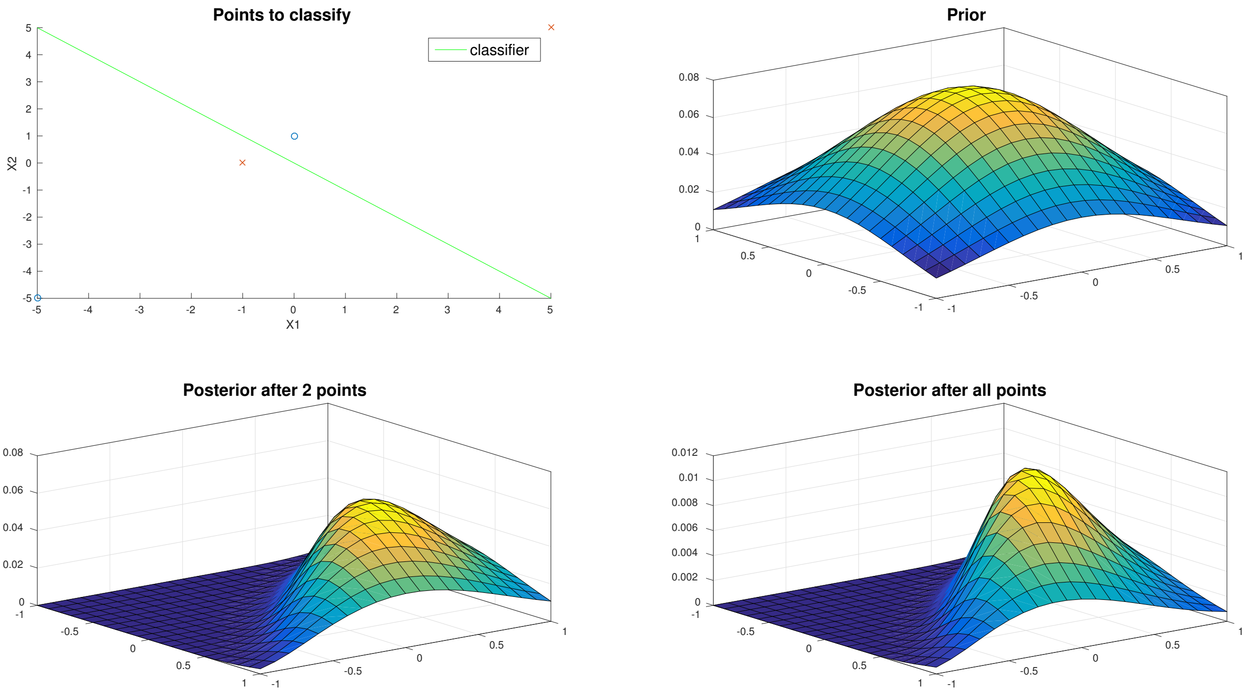

Figure 1: Bottom: evolution of the posterior through training. Top: in green, the classifier extracted using MAP on the weights' distribution

[1] https://github.com/milt0n/ANN-Experiments/bayesnn.m

3 Decision boundary of classifier

In this section, I investigated whether a simple bayesian approach could classify correctly data generated by a perceptron (hence linearly separable). In terms of architecture, the perceptron created consisted of 1 neuron with 2 inputs taking values in $[-1; 1]$, without bias term. The output of the perceptron is in $\{-1, 1\}$

Figure 2 displays the classifiers on top of the dataset (top left-hand side figure). You can see that the classifier generated using the Bayes rule is pretty close to the perceptron used to generate the dataset. There seemed to be exceptions to that, and in a few cases the classifier was off.

For some reason, the convergence of the posterior through the iterative updates always entailed that one of the weight (w1 or w2) was set to 1, while the other varied. I did not see exception to that when running the script multiple times.

The bottom, right-hand side graphic depicts the so-called contour lines of the posterior distribution over the weight space. As mentioned before, the posterior always comprised one of the weight set to 1, and thus the contour plot is always glued to one of the external border/constrains of the weight space. The red asterisk represents the weights of the perceptron. This could take any value in $[-1; 1]$, so it happened during some runs that the 2 did not concord much. However, often the weights of the perceptron were in the vicinity of that of the most probable posterior ones.

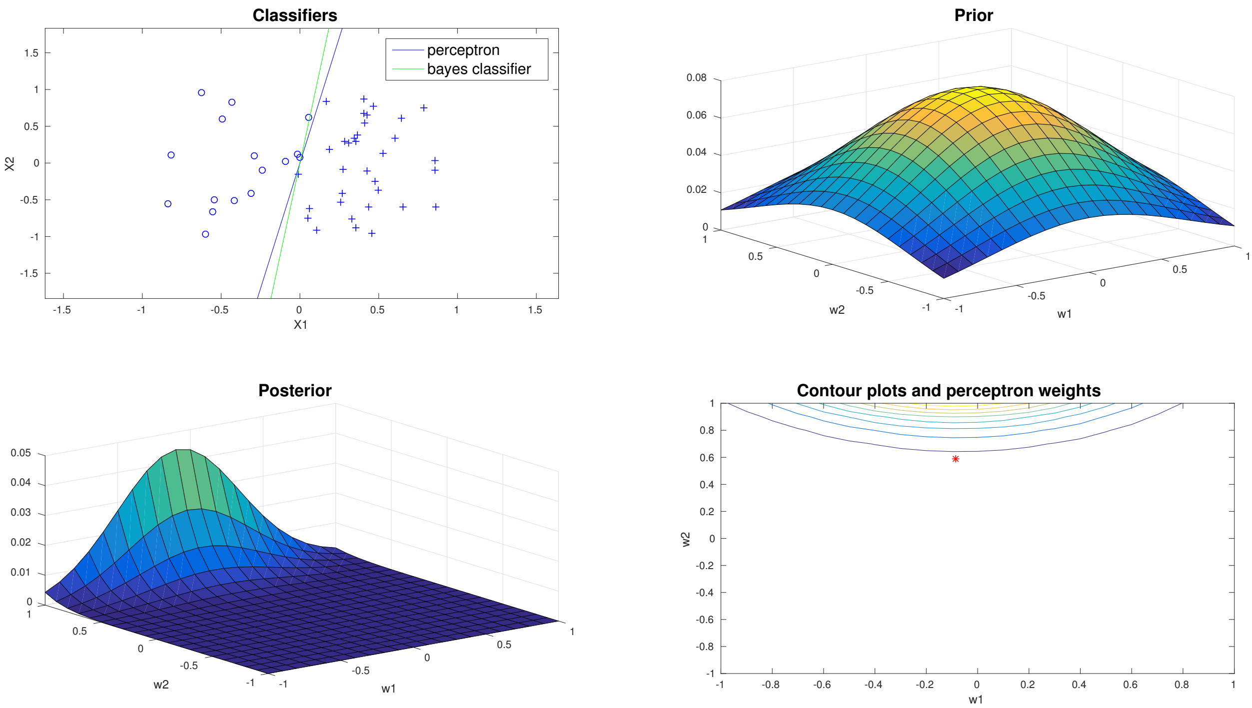

Figure 2: Comparison of perceptron and bayes classifier for dataset generated by perceptron

4 Function approximation

References