Exercises for Artificial Intelligence II

Sean B. Holden, 2013-14

1 Bayesian learning

1. Derive the *weight decay* training algorithm

$$\mathbf{w}_{\text{MAP}} = \underset{\mathbf{w}}{\text{argmin}} \frac{\alpha}{2}||\mathbf{w}||^2 + \frac{\beta}{2}\sum_{i=1}^{m}(y_i - f(\mathbf{w};\mathbf{x}_i))^2$$

   given on slide 268.

2. Use the standard Gaussian integral to derive the final equation for Bayesian regression

$$p(Y|\mathbf{y},\mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma_y^2}}\exp\left(-\frac{(y - f(\mathbf{w}_{\text{MAP}};\mathbf{x}))^2}{2\sigma_y^2}\right)$$

   where

$$\sigma_y^2 = \frac{1}{\beta} + \mathbf{g}^T\mathbf{A}^{-1}\mathbf{g}$$

   given on slide 284.

3. This question asks you to produce a version of the graph on slide 286, but using the Metropolis algorithm instead of the solution obtained by approximating the integral. Any programming language is fine, although Matlab is probably the most straightforward.

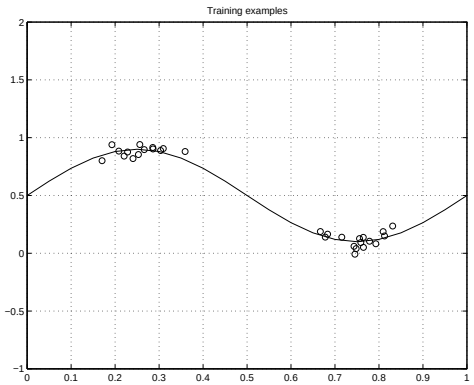
   The data is simple artificial data for a one-input regression problem. Use the target function

$$f(x) = \frac{1}{2} + 0.4\sin 2\pi x$$

   and generate 30 examples clustered around $x = 0.25$ and $x = 0.75$. Then label these examples

$$y(x) = f(x) + n$$

   where n is Gaussian noise of standard deviation 0.05. Plot the data as follows:



Let $\mathbf{w} \in \mathbb{R}^W$ be the vector of all the weights in a network. Your supervised learner should be based on a prior density

$$p(\mathbf{w}) = \left(\frac{2\pi}{\alpha}\right)^{-W/2} \exp\left(-\frac{\alpha}{2}||\mathbf{w}||^2\right)$$

on the weights. A value of $\alpha = 1$ is reasonable. The likelihood used should be

$$p(\mathbf{y}|\mathbf{w}) = \left(\frac{2\pi}{\beta}\right)^{-m/2} \exp\left(-\frac{\beta}{2}\sum_{i=1}^{m}(y(x_i) - h(\mathbf{w}; x_i))^2\right)$$

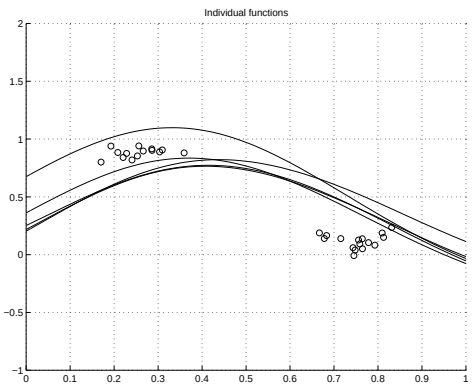
where m is the number of examples and $h(\mathbf{w}; x)$ is the function computed by the neural network with weights $\mathbf{w}$. A value of $\beta = 1/(0.05)^2$ is appropriate. Note that we are assuming that hyperparameters α and β are known, and the prior and likelihood used are the same as those used in the lectures.

Complete the following steps:

(a) Write a function `simpleNetwork` function implementing a multilayer perceptron with a single hidden layer, a basic feedforward structure as illustrated in the AI I lectures, and a single output node. The network should use sigmoid activation functions for the hidden units and a linear activation function for its output. You should use a network having 4 hidden units.

(b) Starting with a weight vector chosen at random, use the Metropolis algorithm to sample the posterior distribution $p(\mathbf{w}|\mathbf{y})$. You should generate a sequence of 100 weight vectors $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{100}$.

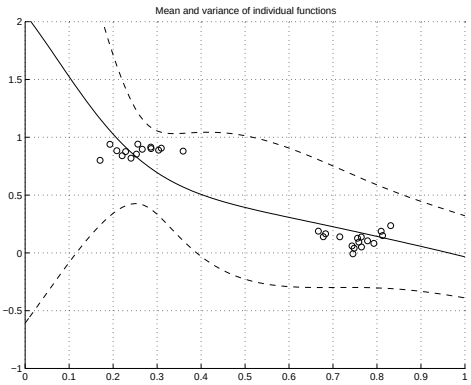
(c) Plot the function $h(\mathbf{w}_i; x)$ computed by the neural network for a few of the weight vectors obtained.



(d) Discard the first 50 weight vectors generated. Using the remainder, calculate the mean and variance of the corresponding functions using

$$\text{mean}(x) = \frac{1}{50}\sum_{i=51}^{100} h(\mathbf{w}_i; x)$$

and a similar expression for the variance. Plot the mean function along with error bars provided by the variance.



4. Can you incorporate hyperparameter estimation into your solution to the previous problem? If so, do the results make sense?

5. Explain how the Gibbs algorithm might be applied to the Bayesian network developed earlier for the *roof-climber alarm* problem.

6. Slide 314 uses the following estimate for the variance of a random variable:

$$\sigma^2 \simeq \hat{\sigma}^2 = \frac{1}{n-1}\left[\sum_{i=1}^{n}(X_i - \hat{X}_n)^2\right].$$

   Show that this estimate is unbiased; that is,

$$\mathbb{E}\left[\hat{\sigma}^2\right] = \sigma^2.$$

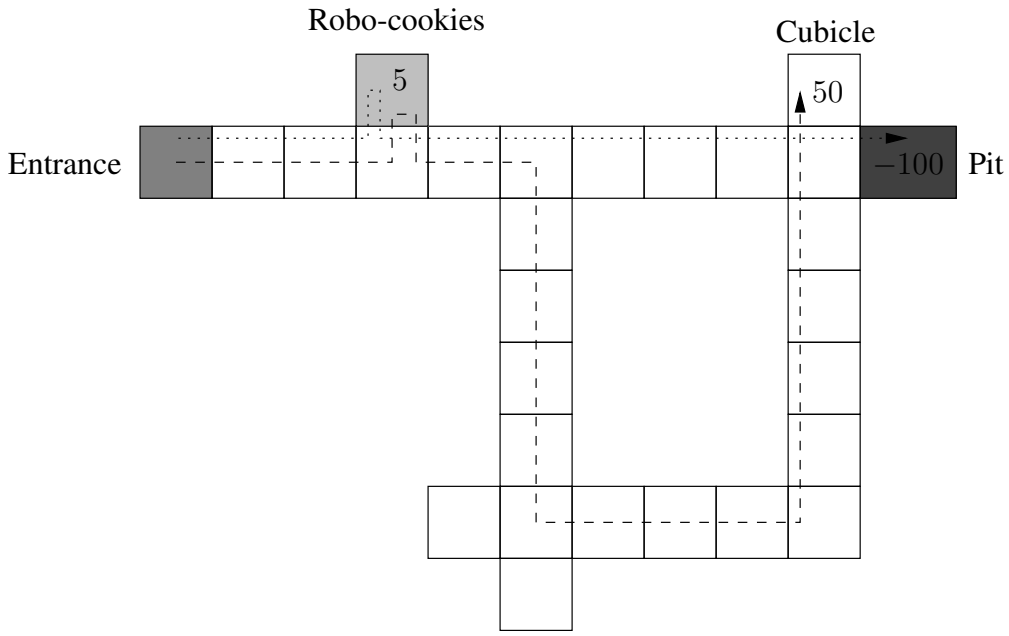
7. Show that if a random variable has zero mean then dividing it by its standard deviation σ results in a new random variable having variance 1.

8. Verify the expression in point 4 on slide 317.

9. Exam question: 2010, paper 8, question 2.

2 Reinforcement learning

1. Evil Robot's Dungeon of Darkness was constructed in such a way that the Pit of Endless Disgruntlement is very close to the Cubicle of Inventive Punishment. Evil Robot likes going to the Cubicle of Inventive Punishment as he gets to be nasty to a human. He does not however like falling into the Pit of Endless Disgruntlement, because that makes him very disgruntled. Between these locations and the entrance is a table with robo-cookies.



Unfortunately the part of his memory related to navigating the Dungeon has accidentally been wiped, so he can't find his way to punish the human responsible. He is running a simulation of the Q-learning algorithm to re-learn it. His actions are to move left, right, up or down one square. Assuming all Q values are initialised to 0, explain how the Q-learning algorithm operates if the dotted route is followed once, then the dashed route is followed, then the dotted route is followed again. Where no reward is indicated assume the reward is 0.

2. Exam question: 2007, paper 9, question 9.

3. Exam question: 2012, paper 7, question 2.