Machine Learning 1 - Homework 4

Available: Monday, October 2nd, 2017
Deadline: Wednesday, 23:59 October 11th, 2017

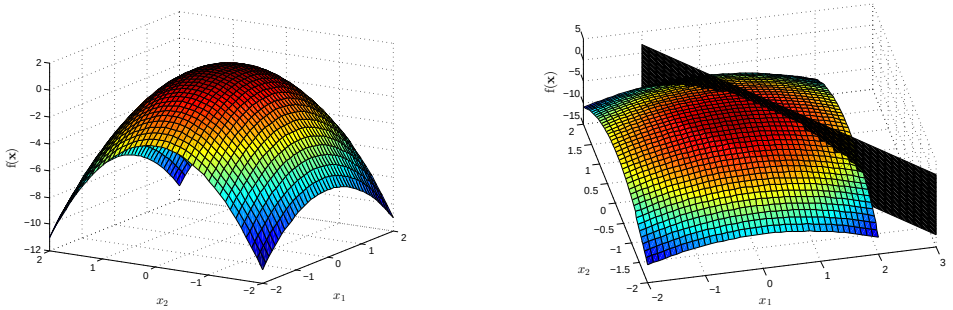
1 Lagrange Multipliers: Warm-up

In this exercise, we will consider optimization problems using Lagrange Multipliers. Suppose we would like to maximize the function

$$f(\mathbf{x}) = 1 - x_1^2 - 2x_2^2$$

which has two input dimensions x_1 and x_2 (they can be considered as the parameters that we would like to learn). This function is plotted in Figure 1(a). We can see the function is concave and there is no local minimum. The optimization of the function is subject to a constraint function. Therefore the optimal solution that is found has to satisfy the constraints.

For example, if we set the constraints $x_1 + x_2 = 1$, then the optimization problem is to find the maximal value of $f(\mathbf{x})$ where $\mathbf{x}$ is also on the constraint plane. Figure 1(b) shows that the constraint function (black) separates $f(\mathbf{x})$ into two parts. The 3D view of $f(\mathbf{x})$ is slightly changed in Figure 1(b) for better visualization.



(a) (b)

Figure 1: Plot (a) is the example function $f(\mathbf{x}) = 1 - x_1^2 - 2x_2^2$. Plot (b) also illustrates the constraint surface.

Answer the following questions below. Note that the first two questions are extra practice questions that will not be graded. They are just meant to provide you with more practice material.

1. **To practice, will not be graded!** Find the maximum of $1-x_1^2-2x_2^2$, subject to the constraint that $x_1 + x_2 = 1$.

2. **To practice, will not be graded!** Find the maximum of $1 - x_1^2 - x_2^2$ subject to the constraint $x_1 + x_2 - 1 \geqslant 0$

3. Find the maximum of $x_1 + 2x_2 - 2x_3$, subject to the constraint that $x_1^2 + x_2^2 + x_3^2 = 1$.

4. Find the maximum of $1-x_1^2-x_2^2$ subject to the constraint $-x_1-x_2+1 \geqslant 0$

5. A company manufactures a chemical product out of two ingredients, known as ingredient X and ingredient Y. The number of doses produced, D, is given by $6x^{2/3}y^{1/2}$, where x and y are the number of grams of ingredients X and Y respectively. Suppose ingredient X costs 4 euro per gram, and ingredient Y costs 3 euro per gram. Find out the maximum number of doses that can be made if no more than 7000 euro can be spent on the ingredients.

2 Kernel Outlier Detection

Consider the picture in Figure 2. The dots represent data-items. Our task is to derive an algorithm that will detect the outliers (in this example there are 2 of them). To that end, we draw a circle rooted at location $\boldsymbol{a}$ and with radius R. All data-cases that fall outside the circle are detected as outliers.

We will now write down the primal program that will find such a circle:

$$\min_{\boldsymbol{a},R,\boldsymbol{\xi}} R^2 + C\sum_{i=1}^{N}\xi_i$$
$$s.t.\ \forall i : \|\boldsymbol{x_i} - \boldsymbol{a}\|^2 \leq R^2 + \xi_i,\ \xi_i \geq 0$$

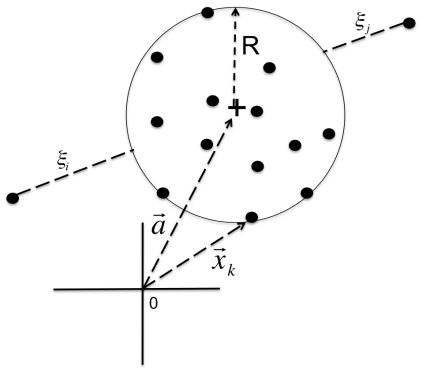


Figure 2: Kernel Outlier Detection

In words: we want to minimize the radius of the circle subject to the constraint that most data-cases should lay inside it. Outliers are allowed to stay outside but they pay a price proportional their distance from the circle boundary and C.

Answer the following questions:

1. Introduce Lagrange multipliers for the constraints and write down the primal Lagrangian. Use the following notation: $\{\alpha_i\}$ are the Lagrange multipliers for the first constraint and $\{\mu_i\}$ for the second constraint.

2. Write down all KKT conditions. (Hint: take the derivative w.r.t. R^2 instead of R).

3. Use these conditions to derive which data-cases $\boldsymbol{x_i}$ will have $\alpha_i > 0$ and which ones will have $\mu_i > 0$.

4. Derive the dual Lagrangian and specify the dual optimization problem. Kernelize the problem, i.e. write the dual program only in terms of kernel entries and Lagrange multipliers.

5. The dual program will return optimal values for $\{\alpha_i\}$. Assume that at least one of these is such that $0 < \alpha_i < C$. In terms of the optimal values for α_i, compute the optimal values for the other dual variables $\{\mu_i\}$.

   Then, solve the primal variables $\{\boldsymbol{a}, R, \boldsymbol{\xi}\}$ (in that order) in terms of the dual variables $\{\mu_i, \alpha_i\}$. Note that you do not need to know the dual optimization program to solve this question. You only need the KKT conditions.

6. Assume we have solved the dual program. We now want to apply it to new test cases. Describe a test in the dual space (i.e. in terms if kernels and Lagrange multipliers) that could serve to detect outliers. (Students who got stuck along the way may describe the test in primal space).

7. What kind of solution do you expect if we use $C = 0$. And what solution if we use $C = \infty$?

8. Describe geometrically what kind of solutions we may expect if we use a RBF kernel (Gaussian) with very small bandwidth (sigma = small), i.e. describe how these solutions can be different geometrically (in x-space) from the case with a linear kernel.

9. Now assume that you are given labels (e.g. y=1 for outlier and y=-1 for "inlier"). Change the primal problem to include these labels and turn it into a classification problem similar to the SVM. (You do not have to derive the dual program).

3 Neural Network

We consider a neural network for binary classification with 2-dimensional inputs $x = (x_1, x_2)^T$, two hiddens neurons and one output, and weights as in Figure 3. Each node (hidden and output) has the logistic sigmoid function as activation function:

$$\sigma(a) = \frac{1}{1 + e^{-a}}.$$

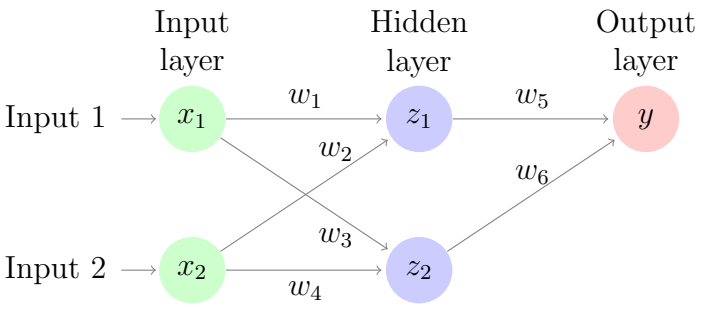


Figure 3: Neural Network

1. The weights are given by $(w_1, w_2, w_3, w_4, w_5, w_6) = (0.4, 0.65, 0.2, 0.1, 0.8, 0.15)$, and we have one data-point $\boldsymbol{x} = (0.1, 0.35)^T$ with target $t = 1$. Do the forward pass to calculate the activations of all nodes and then determine the total error of the neural network.

2. Apply back-propagation to obtain the partial derivatives of the error function with respect to the weights between the hidden layer and the output layer (w_5 and w_6). Then, use these derivatives to update the weights between the hidden layer and the output layer of the network using stochastic gradient descent (SGD) with a learning rate $\eta = 0.05$.

3. Further apply back-propagation to obtain the partial derivatives of the error function with respect to the weights between the input layer and the hidden layer (w_1, w_2, w_3 and w_4). Again, use SGD to find the updates for these weights. Finally, calculate the total error for the neural network with the updated weights.