SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

01.112 Machine Learning, Spring 2018
Homework 1

Due 2 Feb 2018, 5pm

Sample Solutions

Question 1. Consider data points from a 2-d space where each point is of the form $x = (x_1, x_2)$.

(a) You are given a dataset with two positive examples: (1, 1) and (2, 2), and two negative examples (-1, 1) and (1, -1). For each of the following hypothesis spaces, find the parameters of a classifier (a member of the hypothesis space) that can correctly classify all the examples in the dataset, or explain why no such classifier exists.

i. Inside or outside of an origin-centered circle with radius r (r is the parameter). *(5 points)*

ii. Inside or outside of a (a, b)-centered circle with radius r (a, b, r are the parameters). *(5 points)*

iii. Above or below a line through the origin with normal vector $\theta = (\theta_1, \theta_2)$ (or $[\theta_1, \theta_2]^T$). *(5 points)*

(b) Which of the above hypothesis spaces are linear classifiers? *(5 points)*

(a) The training set is ((1, 1), +1), ((2, 2), +1), ((-1, 1), -1), ((1, -1), -1).

i. The decision boundary is given by $x : \|x\| = r$. The classifier h$_r(x)$ is:

$$h_r(x) = \begin{cases} +1, & \text{if } \|x\| > r. \\ -1, & otherwise. \end{cases} \tag{1}$$

Because (1, 1) and (-1, 1) are the same distance from the origin but have opposite labels, $h_r(x)$ will not be able classify the training examples correctly.

ii. Let x_c be the center of the circle with radius r. The decision boundary is given by $\{x : \|x\| = r\}$. A possible classifier h$_{r,x_c}(x)$ can be:

$$h_{r,x_c} = \begin{cases} +1, & \text{if } \|x - x_c\| < r. \\ -1, & otherwise. \end{cases} \tag{2}$$

One example is x_c = (1.5,1.5) and r = 1. h_{r,x_c}. This can classify all examples correctly.

iii. A line through the origin with normal w. The resulting linear decision boundary is defined by $x : w \cdot x = 0$. We can define a classifier $h_w(x)$as:

$$h_w(x) = \begin{cases} +1, & \text{if } w \cdot x \geq 0. \\ -1, & \text{otherwise.} \end{cases} \tag{3}$$

Because the two negative examples belong to the decision boundary normal to $w = (1, 1)$ that includes the origin, it is not possible to separate all training examples accurately according to the above definition of the classifier. However, if you define the classifier as follows:

$$h_w(x) = \begin{cases} +1, & \text{if } w \cdot x > 0. \\ -1, & \text{otherwise.} \end{cases} \tag{4}$$

Then the answer is yes as the two negative points which are exactly on the decision boundary are classified correctly as negative. We will check your answer for understanding and award the points accordingly.

(b) The first two classifiers are non-linear and the last one is linear.

Question 2. Automatic handwritten digit recognition is an important machine learning task. The US Postal Service Zip Code Database (http://www.unitedstateszipcodes.org/zip-code-database/) provides 16 × 16 pixel images preprocessed from scanned handwritten zip codes (US zip codes are the analogues of Singapore postal codes). The task is to recognize the digit in each image. We shall consider the simpler goal of recognizing only two digits: 1 and 5. To simplify our task even further, let's consider only two features: intensity and symmetry. Digit 5 generally occupies more black pixels and thus have higher average pixel intensity than digit 1. Digit 1 is usually symmetric but digit 5 is not. By defining asymmetry as the average difference between an image and its flipped versions, and symmetry as the negation of asymmetry, we can get higher symmetry values for digit 1.

Write an implementation of the perceptron algorithm. Train it on the training set (`train_1_5.csv`), and evaluate its accuracy on the test set (`test_1_5.csv`). The training and test sets are posted on eDimension. csv stands for comma-separated values. In the files, each row is an example. The first value is the symmetry, the second is the average intensity, and the third is the label.

Note: please do NOT shuffle the data. Visit the instances sequentially in the training set when running the perceptron algorithm.

(a) Run the perceptron algorithm for 5 iterations (i.e., traversing the training set 5 times), report the accuracy on the test set. *(5 points)*

(b) Run the perceptron algorithm for 10 iterations, report the accuracy on the test set. *(5 points)*

(c) Submit your code together with crystal clear instructions on how to run it. The TA will follow the instructions to run your code and grade accordingly. *(10 points)*

See the reference implementation