10-315 Introduction to Machine Learning: Homework 4

Due 11:59 p.m. Friday, March 29, 2019

Instructions

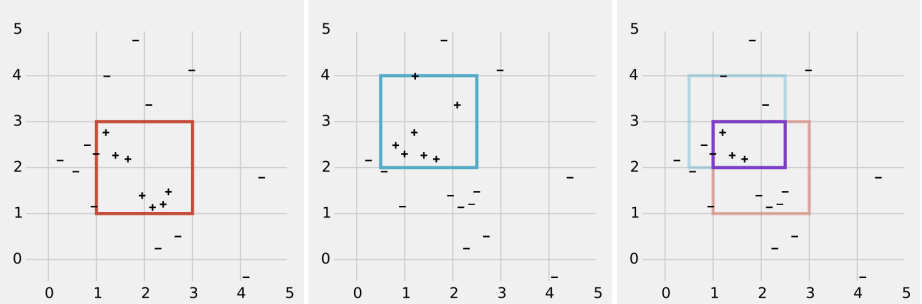
- **Submit your homework on time electronically by submitting to Autolab by 11:59 p.m. Friday, March 29, 2019**. We recommend that you use LaTeX, but we will other typesetting as well. You do not need to download any extra files from Autolab. To submit this homework, you should submit a pdf of your solutions on Autolab by navigating to Homework 4 and clicking the "Submit File" button.
- **Late homework policy**: Homework 4 is worth full credit if submitted before the due date. Up to 50% credit can be received if the submission is less than 48 hours late. The lowest homework grade at the end of the semester will be dropped. Please talk to the instructor in the case of extreme extenuating circumstances.
- **Collaboration policy**: You are welcome to collaborate on any of the questions with anybody you like. However, you must write up your own final solution, and you must list the names of anybody you collaborated with on this assignment.

Problem 1: VC Dimension

Recall that we call a set of points *shattered* by a class of functions H if all possible $\{-1, +1\}$ labelings of the points can be produced by some function in H. The *Vapnik-Chervonenkis* (VC) dimension is the size of the largest set of points that can be shattered by the hypothesis space.

In this problem, we will explore the hypothesis space where each hypothesis is a combination of two simpler hypotheses. More precisely, given two hypotheses h_1 and h_2, we define $h = h_1 \cap h_2$ as a new hypothesis that labels an example $+1$ only if both h_1 and h_2 give the label $+1$, otherwise, it is labeled -1. We can extend this to *sets* of hypotheses: given two sets of hypotheses H_1 and H_2, define $H^* = \{h_1 \cap h_2 : h_1 \in H_1, h_2 \in H_2\}$ as the set of all intersections of hypothesis pairs from the two classes H_1 and H_2.

As an example, let H_1 be the set of classifiers in $\mathbb{R}$ that assigns the label $+1$ if the example is larger than some threshold a. Let H_2 be the set of classifiers in $\mathbb{R}$ that assigns the label $+1$ if the example is smaller than some threshold b. Then H^* would be the set of all intervals (a, b) in $\mathbb{R}$ that assigns $+1$ if the example is inside the interval. Another example is when H_1 and H_2 is the set of all (axis-aligned) squares in $\mathbb{R}^2$, H^* is the set of all axis-aligned rectangles. This example is illustrated below. On the left, we have a single square classifier h_1; in the middle we again have a square classifier h_2; and on the right, we have $h_1 \cap h_2$, which is a rectangle classifier.



Keep in mind that these are only examples. We are looking for results that can apply generally to any pair of hypotheses classes.

1. **[20 pts]** Suppose that the *shattering coefficient* of H_1 is $H_1[n]$ (i.e. the maximum number of ways that the hypothesis space H_1 can label a set of n points is $H_1[n]$). Similarly, suppose that the shattering coefficient of H_2 is $H_2[n]$. Show that $H^*[n] \leq H_1[n]H_2[n]$.

2. For each one of the following function classes, find the VC dimension. State your reasoning.

   i. [10 pts] Half spaces in $\mathbb{R}$, where examples on one side of the boundary are labeled $+1$, and examples on the other side are labeled -1.

   ii. [10 pts] Half spaces in $\mathbb{R}^2$, where examples on one side of the line are labeled $+1$, and examples on the other side are labeled -1.

   iii. [10 pts] Axis-aligned squares in $\mathbb{R}^2$, where points are labeled $+1$ inside the square, and -1 outside (as in the illustrations above).

Problem 2: Linear Regression and Regularization

Suppose that $y = w_0 + w_1 x_1 + w_2 x_2 + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma^2)$. We recommend reading Ch 7.1-7.3 from the Murphy book.

1. **[10 pts]** Write down an expression for $P(y|x_1, x_2)$.

2. **[10 pts]** Assume you are given a set of training observations $(x_1^{(i)}, x_2^{(i)}, y^{(i)})$ for $i = 1, \ldots, n$. Write down the conditional log likelihood of this training data. Drop any constants that do not depend on the parameters w_0, w_1, or w_2.

3. **[20 pts]** Based on your answer above, show that finding the MLE of that conditional log likelihood is equivalent to minimizing least squares, $\frac{1}{2}\sum_{i=1}^{n}(y^{(i)} - (w_0 + w_1 x_1^{(i)} + w_2 x_2^{(i)}))^2$.

4. **[10 pts]** Find the partial derivative of the regularized least squares problem $\frac{1}{2}\sum_{i=1}^{n}(y_i - (w_0 + w_1 x_1^{(i)} + w_2 x_2^{(i)}))^2 + \frac{\lambda}{2}||[w_1, w_2]||_2^2$ with respect to w_0, w_1, and w_2. Although there is a closed form solution to this problem, there are situations in practice where we solve this via gradient descent.

5. **[10 pts extra credit]** Suppose that $w_1, w_2 \sim \mathcal{N}(0, \tau^2)$. Prove that the MAP estimate of w_0, w_1, and w_2 with this prior is equivalent to minimizing the above regularized least squares problem with $\lambda = \frac{\sigma^2}{\tau^2}$.

Implementing Cross Validation (Extra Credit)

In the previous problem, you worked out the gradient of linear least squares with an l_2 norm regularization term on the coefficients w; this is often called Ridge Regression. **In this problem, you will be using cross validation to select the regularization strength, λ, for a related problem called the LASSO (which uses an l_1 norm instead of the l_2 norm)**. Due to the geometry of the l_1 norm, the LASSO encourages the coefficients of w to be sparse (i.e., mostly zero) which can be very helpful for high dimensional datasets.

The general K-Fold Cross Validation algorithm is in Fig. 1. For our implementation: Θ is the set of regularization strengths we are considering, A is the LASSO, and $L_{S_i}(H_{i,\theta})$ is the mean squared error of $h_{i,\theta}$ evaluated on S_i.

You will not have to implement the LASSO yourself and will be using an implementation based on this one. You will implement the following functions from **cv.py**:

1. **[2 pts extra credit]** $get\_fold(X, y, assignments, k)$ returns train/testing split of the data for the requested fold.

2. **[3 pts extra credit]** $evaluate(X\_train, X\_test, y\_train, y\_test, alpha)$ trains and evaluates a LASSO model on the data.

3. **[5 pts extra credit]** $evaluate\_alpha\_by\_cv(X, y, assignments, alpha)$ uses cross validation to evaluate the quality of a LASSO model trained with regularization $alpha$

4. **[2 pts extra credit]** $evaluate\_alphas\_by\_cv(X, y, k, alphas)$ uses cross validation to select the best value of $alpha$ from $alphas$

5. **[8 pts extra credit]** $evaluate\_final\_model(X, y, k, alphas)$ uses cross validation to select the best value of $alpha$ and then evaluates the LASSO model

Documentation and notes about what parameters each function expects, what values it returns, and how to implement it are in the provided code template which can be found on autolab.

k-Fold Cross Validation for Model Selection

input:
training set $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$
set of parameter values Θ
learning algorithm A
integer k
partition S into $S_1, S_2, \ldots, S_k$
foreach $\theta \in \Theta$
for $i = 1 \ldots k$
$h_{i,\theta} = A(S \setminus S_i; \theta)$
$\text{error}(\theta) = \frac{1}{k} \sum_{i=1}^{k} L_{S_i}(h_{i,\theta})$
output
$\theta^\star = \text{argmin}_\theta \left[\text{error}(\theta)\right]$
$h_{\theta^\star} = A(S; \theta^\star)$

Figure 1: The K-Fold Cross Validation algorithm (source)

Note that there is not much code to write, but rather that you must think carefully about how to combine the pieces you have been given. You will submit your code online through the CMU autolab system by uploading **cv.py**, which will execute it remotely against a suite of tests.