CSCI 567, Spring'18
Michael Shindler

Problem Set #3

Due: 11:59 pm, March 18, 2018

Instructions

Submission: Assignment submission will be via courses.uscden.net. By the submission date, there will be a folder set up in which you can submit your files. Please be sure to follow all directions outlined here.

You can submit multiple files, but only the last one submitted counts. That means if you finish some problems and want to submit something now, and then a later file when you finish, that's fine. If I were taking the class, that's what I'd do: that way, if I forget to finish the homework or something happens (remember Murphy's Law), I still get credit for what I finished and turned in. Remember, there are no grace days on problem sets, just on programming assignments!

Problem sets must be typewritten or neatly handwritten when submitted; if the grader cannot read your handwriting on a problem, they may elect to grade it as a zero. If it is handwritten, your submission must still be submitted as a PDF.

It is strongly recommended that you typeset with LaTeX and use that to generate a PDF file.

- The file should be named as `firstname_lastname_USCID.pdf` (e.g., Jenny_Tutone_8675309.pdf).
- Do not have any spaces in your file name when uploading it.
- Please include your name and USCID in the header of the report as well.

There are many free integrated LaTeX editors that are convenient to use. Choose the one(s) you like the most. This http://www.andy-roberts.net/writing/latex seems like a good tutorial.

Collaboration: You may discuss with your classmates. However, you need to write your own solutions and submit separately. Also in your report, you need to list with whom you have discussed for each problem. Please consult the syllabus for what is and is not acceptable collaboration. Review the rules on academic conduct in the syllabus: a single instance of plagiarism can adversely affect you significantly more than you could stand to gain.

Notes on notation:

- Unless stated otherwise, scalars are denoted by small letter in normal font, vectors are denoted by small letters in bold font and matrices are denoted by capital letters in bold font.
- The bias term is subsumed in the input vector, so the input vector is actually $x = [x', 1]^T$, unless mentioned otherwise.
- $\|.\|$ means L2-norm unless specified otherwise i.e. $\|.\| = \|.\|_2$

Problem 1 Kernels (10 points)

In class, we studied kernel functions and their proterties. Consider the following kernel function:

$$k(\mathbf{x}, \mathbf{x}') = \begin{cases} 1, & \text{if } \mathbf{x} = \mathbf{x}' \\ 0, & \text{otherwise} \end{cases}, \quad \forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^D. \tag{1}$$

1.1 Prove that this is a valid kernel. You can apply the Mercer's theorem mentioned in the lecture and assume that the N points $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ you pick are distinct (i.e., $\mathbf{x}_i \neq \mathbf{x}_j$ if $i \neq j$).

1.2 Suppose now you are given a training set $\{(\mathbf{x}_n \in \mathbb{R}^D, y_n \in \mathbb{R})\}_{n=1}^N$ for a regression problem, where $\mathbf{x}_i \neq \mathbf{x}_j$ if $i \neq j$. Show that by using this kernel, training a kernel ridge regressor with $\lambda = 0$ will always lead to the training objective of 0—meaning that all the training examples are *predicted accurately* by the learned regressor. That is, the learned regressor will accurately predict the value of each training example.

The training objective of kernel ridge regression is as follows

$$J(\alpha) = \frac{1}{2}\alpha^T \mathbf{K}^T \mathbf{K} \alpha - \mathbf{y}^T \mathbf{K} \alpha + \frac{\lambda}{2} \alpha \mathbf{K} \alpha + \frac{1}{2} \mathbf{y}^T \mathbf{y}, \tag{2}$$

where $\mathbf{K} \in \mathbb{R}^{N \times N}$ is the kernel matrix, $\mathbf{y} = [y_1, y_2, \cdots, y_N]^T$, and $\alpha \in \mathbb{R}^N$.

The learned regressor is

$$f(\mathbf{x}) = [k(\mathbf{x}, \mathbf{x}_1), k(\mathbf{x}, \mathbf{x}_2), \cdots, k(\mathbf{x}, \mathbf{x}_N)]\alpha^*, \tag{3}$$

where $\alpha^* = \arg\min_\alpha J(\alpha)$

1.3 Although the learned regressor can accurately predict the value of each training example, it does not generalize to the test data. Specifically, show that for any $\mathbf{x}$ with $\mathbf{x} \neq \mathbf{x}_n, \forall n = 1, 2, \cdots, N$, the predicted value is always 0.

Problem 2 Support Vector Machines (20 points)

Consider the dataset consisting of points (x, y), where x is a real value, and $y \in \{-1, 1\}$ is the class label. Let's start with three points $(x_1, y_1) = (-1, -1)$, $(x_2, y_2) = (1, -1)$, $(x_3, y_3) = (0, 1)$.

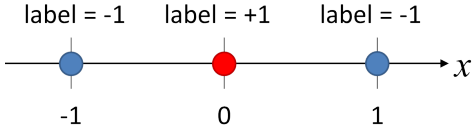

Figure 1: Three data points considered in Problem 2

2.1 Can three points shown in Figure 1, in their current one-dimensional feature space, be perfectly separated with a linear separator? Why or why not?

2.2 Now we define a simple feature mapping $\phi(x) = [x, x^2]^T$ to transform the three points from one- to two-dimensional feature space. Plot the transformed points in the new two-dimensional feature space (use any package you prefer for the plot, e.g., Matplotlib, PowerPoint). Is there a linear decision boundary that can separate the points in this new feature space? Why or why not?

2.3 Given the feature mapping $\phi(x) = [x, x^2]^T$, write down the kernel function $k(x, x')$. Moreover, write down the 3×3 kernel (or Gram) matrix $\mathbf{K}$ based on $k(x_i, x_j)$ of the three data points. Verify that $\mathbf{K}$ is a positive semi-definite (PSD) matrix. You may want to show this by the definition of PSD matrix: a symmetric $N \times N$ real matrix $\mathbf{M}$ is said to be positive semi-definite if the scalar $\mathbf{z}^T\mathbf{M}\mathbf{z}$ is non-negative for every non-zero column vector $\mathbf{z}$ of N real numbers.

2.4 Now you are going to learn a *hard margin* SVM classifier on this dataset with $k(x_i, x_j)$. Recall the primal and dual formulation you learned in lectures. Write down the primal and dual formulations of this problem.

2.5 Next, you are going to solve this problem using its dual formulation. You may want to use the symmetric property to simplify the dual formulation.

2.6 Let $\hat{y} = \mathbf{w}^T\phi(\mathbf{x}) + b$, where $\mathbf{w}$ and b are the weights and the bias you got from the previous question. Draw the decision boundary in the new two-dimensional feature space and circle all support vectors. (Set $\hat{y}$ to 0 to get the decision boundary). Then, draw the decision boundary in the original one-dimensional setting.

Problem 3 Decision trees

Question 3.1 through 3.3 deal with the following data. Given a dataset with two classes, we have 200 samples in class A and 200 samples in class B. Now we are given two tree models α and β.

At one leaf node of α there are 150 samples in class A and 50 samples in class B, and the other leaf node has 50 samples in class A and 150 samples in class B. One leaf node of β has 0 sample in class A and 100 samples in class B, and the other leaf node of β has 200 in class A and 100 in class B.

3.1 For each leaf node given above, compute the corresponding misclassification rate, cross-entropy and Gini index.

3.2 Compare the trees given above using misclassification rate, for which you compare the total number of misclassified samples of the tree over the total number of samples. Which tree do you favor and why?

3.3 Given the impurity measure $Q_T(m)$ for the leaf m of tree T, let's define the tree comparison measure as

$$C_T = \left(\sum_m Q_T(m) \right) + \lambda |T|, \tag{4}$$

where λ is a regularization paramter and $|T|$ measures the number of leaf nodes.

Compare the trees given above using cross-entropy and Gini index. Which tree do you favor and why?

3.4 Under the regression setting, we would like to use decision tree to predict the output value of a given sample.

Define the sum-of-square error for region R_m as

$$cost(\{(\mathbf{x}_n, y_n) : \mathbf{x}_n \in R_m\}) = \sum_{\{n:\mathbf{x}_n \in R_m\}} (y_n - \tilde{y})^2 \tag{5}$$

Show that given a tree model with the partition of the input space, by minimizing the sum-of-square error in the corresponding region, the predicted value $\tilde{y}$ equals to the average of all data points within that region.

Problem 4 Boosting

4.1 In AdaBoost, we minimize the exponential loss at step t:

$$L_t = (e^{-\beta_t} + e^{\beta_t}) \sum_n w_t(n) \mathbb{I}[y_n \neq h_t(\mathbf{x}_n)] + e^{-\beta_t} \sum_n w_t(n) \tag{6}$$

Show that the parameter β_t are updated using

$$\beta_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}, \tag{7}$$

where

$$\epsilon_t = \frac{\sum_n w_t(n) \mathbb{I}[y_n \neq h_t(\mathbf{x}_n)]}{\sum_n w_t(n)} \tag{8}$$

4.2 Show that the exponential error function

$$\epsilon_{exp} = \sum_n e^{-y_n f(\mathbf{x}_n)} \tag{9}$$

does not correspond to the log likelihood of any well-behaved probabilistic model. (Hint: first assume it as the log likelihood of some probabilistic model, then show that the corresponding conditional distribution $p(y|x)$ cannot be correctly normalized)

4.3 As an alternative, the logloss which does correspond to the log likelihood of a probabilistic model, is used in LogitBoost. Show the corresponding probability distribution $p(y_n|\mathbf{x}_n)$ of the logloss function

$$\epsilon_{log} = \sum_n \log(1 + e^{-2y_n f(\mathbf{x}_n)}). \tag{10}$$

is

$$p(y_n|\mathbf{x}_n) = \frac{e^{y_n f(\mathbf{x}_n)}}{e^{y_n f(\mathbf{x}_n)} + e^{-y_n f(\mathbf{x}_n)}} \tag{11}$$