CSCI 567, Spring'18
Michael Shindler

Problem Set #3

Due: 11:59 pm, March 9, 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 $i \neq j$).

If the N points $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ are distinct, the kernel matrix $\mathbf{K}$ is an N-by-N identity matrix, which is positive (semi-) definite. Then according to Mercer's theorem, $k(\mathbf{x}, \mathbf{x}')$ is a valid kernel function.

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

When $\lambda = 0$, the objective becomes

$$\begin{aligned}
J(\alpha) &= \frac{1}{2}\alpha^T \mathbf{K}^T \mathbf{K}\alpha - \mathbf{y}^T \mathbf{K}\alpha + \frac{1}{2}\mathbf{y}^T \mathbf{y} \\
&= \frac{1}{2}||\mathbf{K}\alpha - \mathbf{y}||_2^2 \qquad \text{(This is the sum of the prediction error of training examples.)} \\
&= \frac{1}{2}||\mathbf{I}\alpha - \mathbf{y}||_2^2 \\
&= \frac{1}{2}||\alpha - \mathbf{y}||_2^2 \\
&= 0 \text{ if } \alpha = \mathbf{y}. \qquad \text{(0 is the minimum achievable value of } J(\alpha)\text{; i.e., perfect prediction.)}
\end{aligned}$$

Thus, $\alpha^* = \mathbf{y}$.

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.

For any $\mathbf{x}$ with $\mathbf{x} \neq \mathbf{x}_n, \forall n = 1, 2, \cdots, N$, the predicted value is

$$f(\mathbf{x}) = [k(\mathbf{x}, \mathbf{x}_1), k(\mathbf{x}, \mathbf{x}_2), \cdots, k(\mathbf{x}, \mathbf{x}_N)]\alpha^* = \mathbf{0}^T \mathbf{y} = 0.$$

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 $(x_1, y_1) = (-1, -1)$, $(x_3, y_3) = (0, 1)$, $(x_2, y_2) = (1, -1)$.

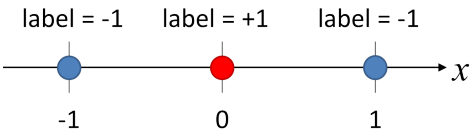

Figure 1: Three data points considered in Problem 2

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

The dataset is non-linear seperatable. For any point on the real line, the dataset cannot be currectly separated.

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?

Yes. Any horizonal line with inception between 0 and 1 can correctly seperate the data.

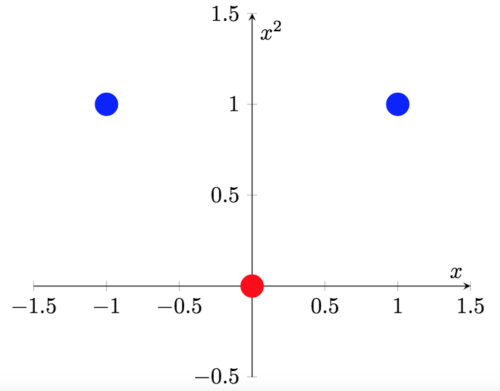

Figure 2: Plot for Q2.2: data points in new 2D space

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.

$$\mathbf{K} = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4}$$

$$\tag{5}$$

$$\forall \mathbf{z} \in \mathbb{R}^3, \qquad \mathbf{z}^T\mathbf{K}\mathbf{z} = 2z_1^2 + 2z_2^2 \geq 0 \tag{6}$$

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.

Primal Formulation:

$$\min_{\mathbf{w},b} \quad \frac{1}{2}||\mathbf{w}||_2^2 \tag{7}$$

$$\text{s.t.} \quad 1 - y_n[\mathbf{w}^T\phi(\mathbf{w}_n) + b] \leq 0, \forall n \tag{8}$$

Dual Formulation:

$$\max_{\alpha} \quad \sum_n \alpha_n - \frac{1}{2}\sum_{m,n} y_m y_n \alpha_m \alpha_n k(\mathbf{x}_m, \mathbf{x}_n) \tag{9}$$

$$\text{s.t.} \quad 0 \leq \alpha_n \leq +\infty, \forall n \tag{10}$$

$$\sum_n \alpha_n y_n = 0 \tag{11}$$

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.

From symmetric, $\alpha_1 = \alpha_2$.

$$\max_{\alpha} \quad \sum_n \alpha_n - \frac{1}{2}\sum_{m,n} y_m y_n \alpha_m \alpha_n k(\mathbf{x}_m, \mathbf{x}_n) \tag{12}$$

$$\text{s.t.} \quad 0 \leq \alpha_1, \alpha_3 \tag{13}$$

$$\alpha_3 = 2\alpha_1 \tag{14}$$

Plugging $\mathbf{K}$ we get

$$\alpha^\star = \arg\min_{\alpha} \alpha_1^2 + \alpha_2^2 - \alpha_1 - \alpha_2 - \alpha_3 \tag{15}$$

$$= \arg\min_{\alpha} 2\alpha_1^2 - 4\alpha_1 \tag{16}$$

$$\Rightarrow \quad \alpha^\star = (1, 1, 2)^T \tag{17}$$

Then

$$\mathbf{w} = \sum_n y_n \alpha_n \phi(\mathbf{x}_n) = (0, -2)^T \tag{18}$$

$$b = y_n - \mathbf{w}^T\phi(\mathbf{x}_n) = 1 \tag{19}$$

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.

For decision boundary in new 2D space:

$$\text{Set} \quad 0 = \hat{y} = \mathbf{w}^T\phi(\mathbf{x}) + b = (0, -2)(x, y)^T + 1 \tag{20}$$

$$\Rightarrow \quad y = \frac{1}{2} \tag{21}$$

For decision boundary in orignial 1D space:

$$\text{Set} \quad 0 = \hat{y} = \mathbf{w}^T \phi(\mathbf{x}) + b = (0, -2)(x, x^2)^T + 1 \tag{22}$$

$$\Rightarrow \quad x = \pm \frac{\sqrt{2}}{2} \tag{23}$$

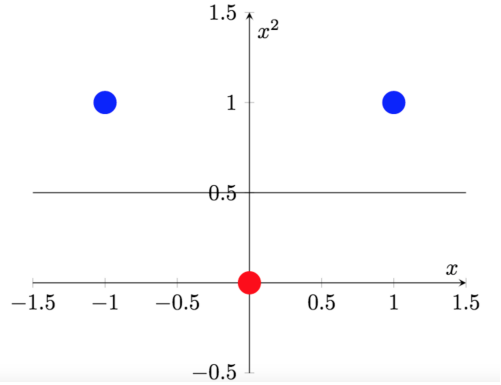

Figure 3: Plot for Q2.6: decision boundary in 2D space

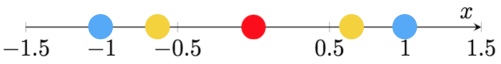

Figure 4: Plot for Q2.6: decision boundary in 1D space

Problem 3 Decision trees

Question **??** through **??** 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.

Misclassification rates:

$$MR_{\alpha_1} = \frac{50}{150+50} = 0.25$$
$$MR_{\alpha_2} = \frac{50}{50+150} = 0.25$$
$$MR_{\beta_1} = \frac{0}{0+100} = 0$$
$$MR_{\beta_2} = \frac{100}{200+100} \approx 0.33$$

Cross-entropy:

$$CE_{\alpha_1} = -\frac{150}{150+50}\log(\frac{150}{150+50}) - \frac{50}{150+50}\log(\frac{50}{150+50}) \approx 0.56$$
$$CE_{\alpha_2} = -\frac{50}{150+50}\log(\frac{50}{150+50}) - \frac{150}{150+50}\log(\frac{150}{150+50}) \approx 0.56$$
$$CE_{\beta_1} = -\frac{0}{0+100}\log(\frac{0}{0+100}) - \frac{100}{0+100}\log(\frac{100}{0+100}) = 0$$
$$CE_{\beta_2} = -\frac{200}{200+100}\log(\frac{200}{200+100}) - \frac{100}{100+200}\log(\frac{200}{200+100}) \approx 0.64$$

Gini index:

$$GI_{\alpha_1} = 1 - (\frac{150}{150+50})^2 - (\frac{50}{150+50})^2 = 0.375 \approx 0.38$$
$$GI_{\alpha_2} = 1 - (\frac{50}{150+50})^2 - (\frac{150}{150+50})^2 = 0.375 \approx 0.38$$
$$GI_{\beta_1} = 1 - (\frac{0}{0+100})^2 - (\frac{100}{0+100})^2 = 0$$
$$GI_{\beta_2} = 1 - (\frac{200}{200+100})^2 - (\frac{100}{200+100})^2 \approx 0.44$$

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?

$$MR_{\alpha} = \frac{50+50}{200+200} = 0.25$$
$$MR_{\beta} = \frac{0+100}{200+200} = 0.25$$

They have the same misclassification rate.

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

$$C_T = \left(\sum_m Q_T(m) \right) + \lambda |T|, \tag{24}$$

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?

Cross-entropy:

$$C_{\alpha_{CE}} = CE_{\alpha_1} + CE_{\alpha_2} + 2\lambda \approx 1.12 + 2\lambda$$
$$C_{\beta_{CE}} = CE_{\beta_1} + CE_{\beta_2} + 2\lambda \approx 0.64 + 2\lambda$$

Gini index:

$$C_{\alpha_{GI}} = GI_{\alpha_1} + GI_{\alpha_2} + 2\lambda = 0.75 + 2\lambda$$
$$C_{\beta_{GI}} = GI_{\beta_1} + GI_{\beta_2} + 2\lambda \approx 0.44 + 2\lambda$$

Tree β has lower impurity.

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

$$cost(\{(\mathbf{x}_n, y_n) : \mathbf{x}_n \in R_m\}) = \sum_{\{n:\mathbf{x}_n \in R_m\}} (y_n - \tilde{y})^2 \tag{25}$$

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.

$$\frac{\partial \sum_{\{n:\mathbf{x}_n \in R_m\}} (y_n - \tilde{y})^2}{\partial \tilde{y}} = \sum_{\{n:\mathbf{x}_n \in R_m\}} 2(\tilde{y} - y_n)$$

By setting the partial derivative to zero, we have

$$\tilde{y} = \frac{1}{N_m} \sum_{\{n:\mathbf{x}_n \in R_m\}} y_n,$$

where N_m is the number of samples in region m.

Problem 4 Boosting

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

$$L_t = (e^{\beta_t} - e^{-\beta_t}) \sum_n w_t(n) \mathbb{I}[y_n \neq h_t(\mathbf{x}_n)] + e^{-\beta_t} \sum_n w_t(n) \tag{26}$$

Show that the parameter β_t are updated using

$$\beta_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}, \tag{27}$$

where

$$\epsilon_t = \frac{\sum_n w_t(n)\mathbb{I}[y_n \neq h_t(\mathbf{x}_n)]}{\sum_n w_t(n)} \tag{28}$$

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

By setting the partial derivative to zero, we have

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

Then we get

$$\epsilon_t = \frac{\sum_n w_t(n)\mathbb{I}[y_n \neq h_t(\mathbf{x}_n)]}{\sum_n w_t(n)} = \frac{e^{-\beta_t}}{e^{\beta_t} + e^{-\beta_t}} = \frac{1}{e^{2\beta_t} + 1}$$

4.2 Show that the exponential error function

$$\epsilon_{exp} = \sum_n e^{-y_n f(\mathbf{x}_n)} \tag{29}$$

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)

Assume that the exponential error function corresponds to a valid log likelihood, then its corresponding likelihood function is

$$e^{-\epsilon_{exp}} = \prod_n e^{-e^{-y_n f(\mathbf{x}_n)}}.$$

Therefore we have

$$p(y_n|\mathbf{x}_n) \propto e^{-e^{-y_n f(\mathbf{x}_n)}}.$$

This probability distribution can be normalized by

$$Z = e^{-e^{f(\mathbf{x}_n)}} + e^{-e^{-f(\mathbf{x}_n)}},$$

which involves $f(\mathbf{x}_n)$. After applying the normalization factor Z, it becomes

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

In this case, the error function obtained from negative log-likelihood is different from (**??**).

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

$$\epsilon_{log} = \sum_n \log(1 + e^{-2y_n f(\mathbf{x}_n)}). \tag{30}$$

is

$$p(y_n|\mathbf{x}_n) = \frac{e^{y_n f(\mathbf{x}_n)}}{e^{y_n f(\mathbf{x}_n)} + e^{-y_n f(\mathbf{x}_n)}} \tag{31}$$

The corresponding likelihood function is

$$e^{-\epsilon_{log}} = \prod_n \frac{1}{1 + e^{-2y_n f(\mathbf{x}_n)}}$$

Therefore we have

$$p(y_n|\mathbf{x}_n) = \frac{1}{1 + e^{-2y_n f(\mathbf{x}_n)}} = \frac{e^{y_n f(\mathbf{x}_n)}}{e^{y_n f(\mathbf{x}_n)} + e^{-y_n f(\mathbf{x}_n)}}.$$