CS 446 / ECE 449 — Homework 3

scs13

Version 1.0

Instructions.

- Homework is due **Thursday, March 3, at noon CST**; no late homework accepted.
- Everyone must submit individually on Gradescope under `hw3` and `hw3code`. Problem parts are marked with [`hw3`] and [`hw3code`] to indicate where they are handed in.
- The "written" submission at `hw3` **must be typed**, and submitted in any format Gradescope accepts (to be safe, submit a PDF). You may use LaTeX, Markdown, Google Docs, MS Word, whatever you like; but it must be typed!
- When submitting at `hw3`, Gradescope will ask you to select pages for each problem; please do this precisely!
- Please make sure your NetID is clear and large on the first page of the homework.
- Your solution **must** be written in your own words. Please see the course webpage for full academic integrity information. Briefly, you may have high-level discussions with at most 3 classmates, whose NetIDs you should place on the first page of your solutions, and you should cite any external reference you use; despite all this, your solution must be written in your own words.
- We reserve the right to reduce the auto-graded score for `hw3code` if we detect funny business (e.g., your solution lacks any algorithm and hard-codes answers you obtained from someone else, or simply via trial-and-error with the autograder).
- Coding problems come with suggested "library routines"; we include these to reduce your time fishing around APIs, but you are free to use other APIs.
- When submitting to `hw3code`, upload `hw3.py`. Don't upload a zip file or additional files.

Version history.

1.0. Initial version.

1. ResNet.

In this problem, you will implement a simplified ResNet. You do not need to change arguments which are not mentioned here (but you of course could try and see what happens).

(a) [hw3code] Implement a class `Block`, which is a building block of ResNet. It is described in Figure 2 of He et al. (2016), but also as follows.

The input to `Block` is of shape (N, C, H, W), where N denotes the batch size, C denotes the number of channels, and H and W are the height and width of each channel. For each data example $\boldsymbol{x}$ with shape (C, H, W), the output of `block` is

$$\texttt{Block}(\boldsymbol{x}) = \sigma_r\left(\boldsymbol{x} + f(\boldsymbol{x})\right),$$

where σ_r denotes the ReLU activation, and $f(\boldsymbol{x})$ also has shape (C, H, W) and thus can be added to $\boldsymbol{x}$. In detail, f contains the following layers.

i. A `Conv2d` with C input channels, C output channels, kernel size 3, stride 1, padding 1, and no bias term.
ii. A `BatchNorm2d` with C features.
iii. A ReLU layer.
iv. Another `Conv2d` with the same arguments as i above.
v. Another `BatchNorm2d` with C features.

Because 3×3 kernels and padding 1 are used, the convolutional layers do not change the shape of each channel. Moreover, the number of channels are also kept unchanged. Therefore $f(\boldsymbol{x})$ does have the same shape as $\boldsymbol{x}$.

Additional instructions are given in doscstrings in `hw3.py`.

Library routines: `torch.nn.Conv2d and torch.nn.BatchNorm2d.`

Remark: Use `bias=False` for the `Conv2d` layers.

(b) [hw3code] Implement a (shallow) `ResNet` consists of the following parts:

i. A `Conv2d` with 1 input channel, C output channels, kernel size 3, stride 2, padding 1, and no bias term.
ii. A `BatchNorm2d` with C features.
iii. A ReLU layer.
iv. A `MaxPool2d` with kernel size 2.
v. A `Block` with C channels.
vi. An `AdaptiveAvgPool2d` which for each channel takes the average of all elements.
vii. A `Linear` with C inputs and 10 outputs.

Additional instructions are given in doscstrings in `hw3.py`.

Library routines: `torch.nn.Conv2d, torch.nn.BatchNorm2d, torch.nn.MaxPool2D, torch.nn.AdaptiveAvgPool2d and torch.nn.Linear.`

Remark: Use `bias=False` for the `Conv2d` layer.

(c) [hw3] Train your `ResNet` implemented in (b) with different choices $C \in \{1, 2, 4\}$ on digits data and draw the training error vs the test error curves. To make your life easier, we provide you with the starter code to load the digits data and draw the figures with different choices for C. Therefore, you only need to write the code to train your `ResNet` in function `plot_resnet_loss_1()`. See the docstrings in `hw3.py` for more details. Include the resulting plot in your written handin.

For full credit, in addition to including the six train and test curves, include at least one complete sentence describing how the train and test error (and in particular their gap) change with C, which itself corresponds to a notion of model complexity as discussed in lecture.

Library routines: `torch.nn.CrossEntropyLoss, torch.autograd.backward, torch.no_grad, torch.optim.Optimizer.zero_grad, torch.autograd.grad, torch.nn.Module.parameters.`

(d) [hw3] Train your `ResNet` implemented in (b) with $C = 64$ on digits data and draw the training error vs the test error curve. To make your life easier, we provide you with the starter code to load the digits data and draw the figures with $C = 64$. Therefore, you only need to write the code to train your `ResNet` in function `plot_resnet_loss_2()`. See the docstrings in `hw3.py` for more details. Notice that you can use the same implementation of training part in part (c). Include the resulting plot in your written handin.

For full credit, additionally include at least one complete sentence comparing the train and test error with those in part (c).

Library routines: `torch.nn.CrossEntropyLoss, torch.autograd.backward, torch.no_grad, torch.optim.Optimizer.zero_grad, torch.autograd.grad, torch.nn.Module.parameters.`

Solution.

(a) The Block class has been implemented according to the layers mentioned.
(b) The shallow ResNet has been implemented as mentioned in the question.
(c) The result obtained on training ResNet with C ∈ 1,2,4 is as shown below:

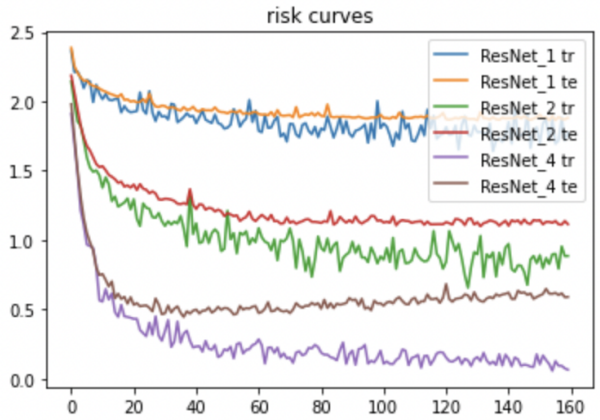


Figure 1: Output plot after training with C ∈ 1,2,4

From the plot obtained, it can be seen that as the value of C increases, the decrease in risk value through the training drastically increases (i.e., C = 4 has a greater dip in risk compared to C = 1 and C = 2). Also, test curve is above training curve. As theoretically visible, the risk on training set is lesser than the risk of test set. Additionally, the gap between train and test risk curves seem to be increasing as C increases, but the close proximity of C values inhibits from reaching the conclusion that this difference is statistically significant.

(d) The result obtained on training ResNet with C $\in$ 1,2,4 is as shown below:

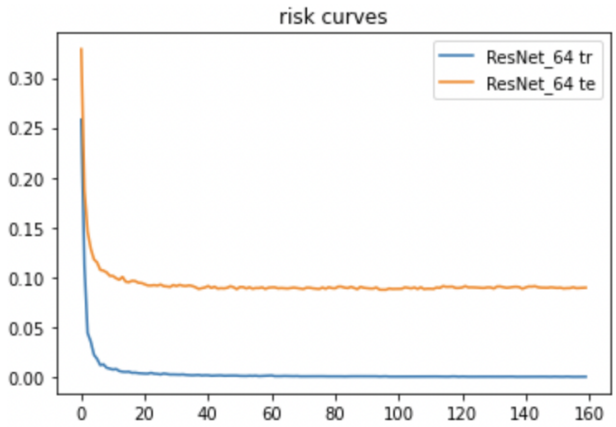
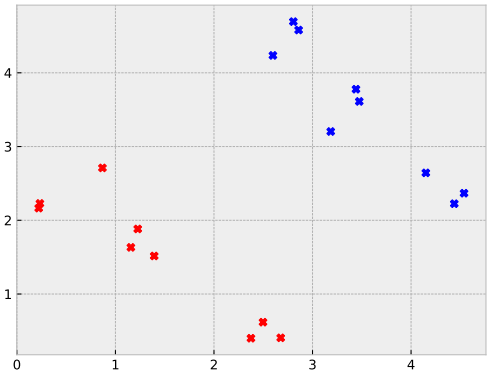


Figure 2: Output plot after training with C = 64

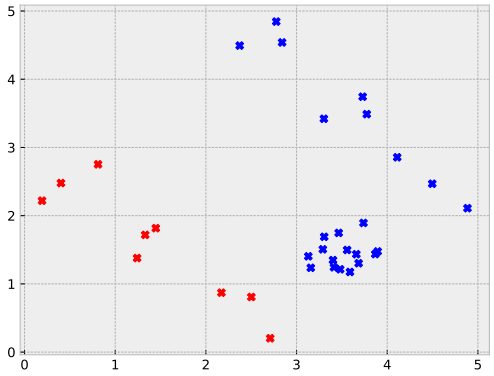
The plot obtained in (d) shows greater gap between training and testing set than the plots obtained in (c). This can be confirmed by taking a simple difference of the x-axis values in both curves. Again, whether this difference is a statistically significant outcome can be determined on the basis of confidence interval requirements. The risk curve however, is smoother here as compared to those in (c).

2. Decision Trees and Nearest Neighbor.

Consider the training and testing data sets as given in Figures 3a and 3b for the sub-parts (a)-(c). For the sub-parts (d)-(f), refer to the training and testing data sets as given in Figures 9a and 9b. For problems related to decision trees, either draw the decision trees unambiguously on the figure (e.g., either with drawing software, or by taking a picture of a hand drawing) and include the modified diagrams in your submission, or use unambiguous pseudocode to specify the decision trees.



(a) Fig 1a: Training data set 1.

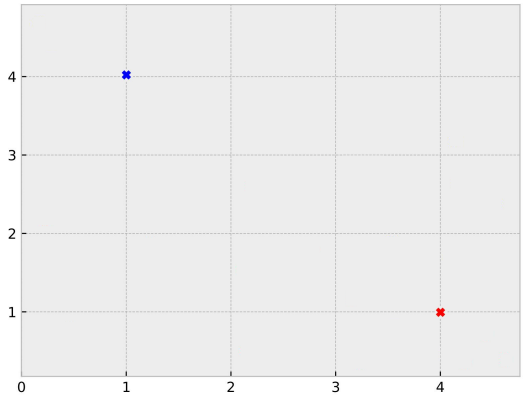


(b) Fig 1b: Testing data set 1.

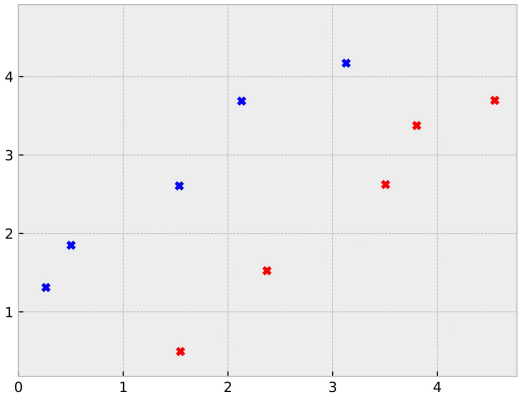
(a) [hw3] Define in pseudocode or draw (as above) a decision tree of depth one with integral and axis-aligned decision boundaries which achieves error at most $\frac{1}{6}$ on training data set 1 (Figure 3a).
Note: "integral and axis-aligned" means the decision tree consists of splitting rules of the form $[x_1 \geq 5]$, $[x_2 < 3]$, and so on.

(b) [hw3] Define in pseudocode or draw (as above) a decision tree (of any depth) with integral and axis-aligned decision boundaries which achieves zero error on training data set 1 (Figure 3a).

(c) [hw3] Define in pseudocode or draw (as above) a decision tree (of any depth) with integral and axis-aligned decision boundaries which achieves zero error on training data set 1 (Figure 3a) but has error at least $\frac{1}{4}$ on testing data set 1 (Figure 3b).



(a) Fig 2a: Training data set 2.



(b) Fig 2b: Testing data set 2.

(d) [hw3] Define in pseudocode or draw (as above) a decision tree with integral and axis-aligned decision boundaries with at most two splits, which achieves zero error on training data set 2 (Figure 9a) and calculate its error on testing data set 2 (Figure 9b).

(e) [hw3] Construct and draw a 1-nearest-neighbor classifier using training data set 2 (Figure 9a). Then copy over that classifier to the corresponding testing data set 2 (Figure 9b). As discussed in class, the training error will be zero; what is the test error on testing data set 2 (Figure 9b)? For full points, include both figures and at least one complete sentence stating the test error.

(f) [hw3] Comparing the result of the decision tree from part (d) and the result of the 1-nn classifier from part (e). Which one has a smaller training error? Which one has a smaller test error? Which algorithm is more suitable here? (In case that both algorithms have the same error, state that they have the same error.)

Solution. Here, x_1 represents the X-axis and x_2 represents the Y-axis.

(a) There are multiple boundaries that fit the requirements - $x_1 > 2$, $x_1 > 3$, $x_2 > 2$ and $x_2 > 3$. The image below shows: $x_1 > 2$

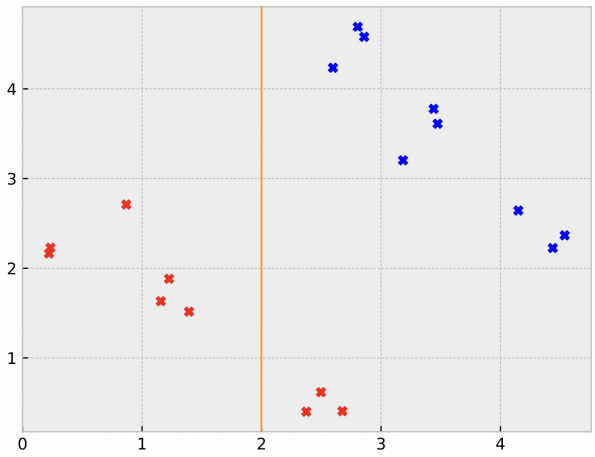


Figure 5: Training data set 1.

There are some misclassified points as shown. Let category 1 data points be the red ones and category 2 datapoints be the blue ones. Three red points are mis-classified out of a total of 18 points in the training set. The error is thus $\frac{3}{18} = \frac{1}{6}$. This satisfies the condition in the question.

(b) To obtain zero training error, the above decision boundary can be further split with integral and axis-aligned boundary condition: $x_2 > 2$ to obtain zero training error. The image is as shown below:

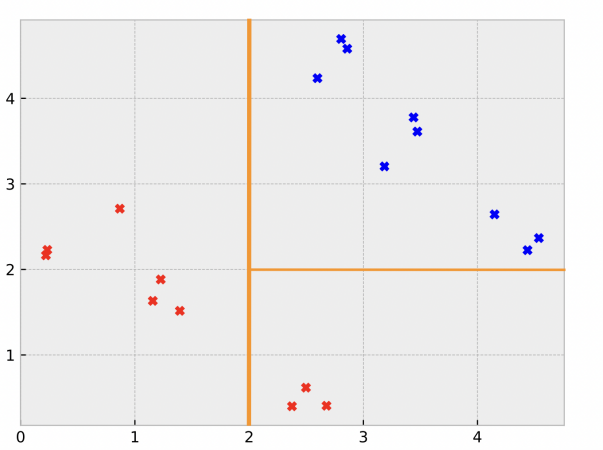
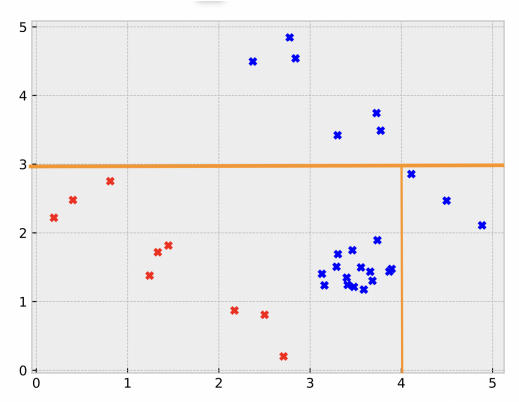
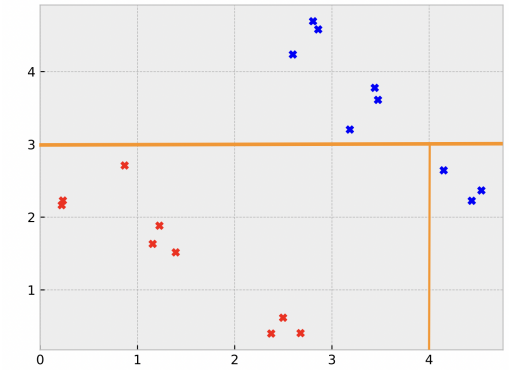


Figure 6: Training data set 1 with zero error.

(c) The following decision boundary with two splits gives the required result with zero training error and more than $\frac{1}{4}$ error. In specific, there are 33 points out of which 15 of them in the test set wrongly classified. The error is $\frac{15}{33}(> \frac{1}{4})$. The boundaries are - $x_2 > 3$ and $x_1 > 4$. It appears as shown below:

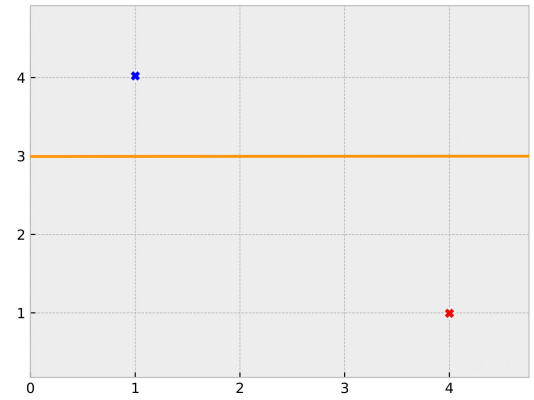


(a) Training data set 1 with zero error.

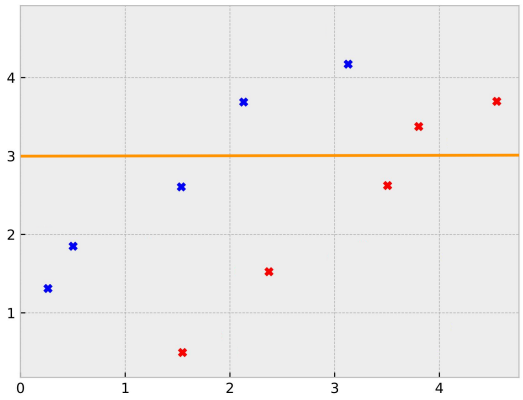


(b) Testing data set 1 with error > 0.25.

(d) To split the training error, a decision boundary: $x_2 > 3$ was used. The diagram looks as shown below on training and testing sets:



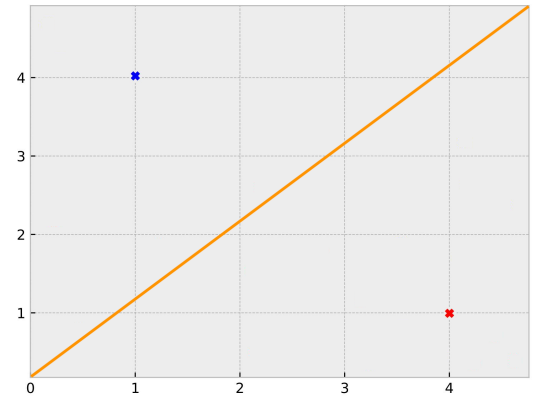
(a) Training data set 2 with zero error.



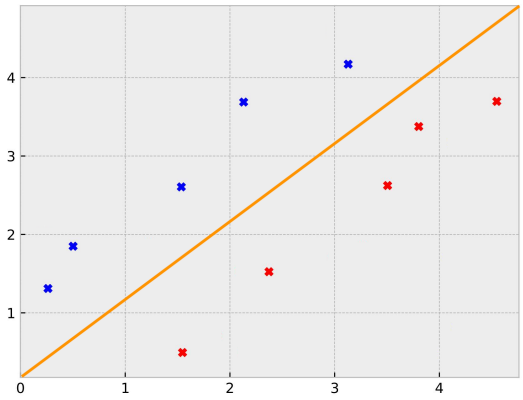
(b) Testing data set 2 with error $\frac{5}{10}$.

The error for training set is zero. The error on the test set is given by $\frac{5}{10}$ - since there are 5 mis-classified points (2 red points in area $x_2 > 3$ and three red points in $x_2 < 3$)

(e) To construct a 1-nn classifier, a simple partition $x_1 = x_2$ is used. The effect of this partition on training set and testing set are as shown below: The error in case of training and testing sets is zero.



(a) Training data set 2 with 1-nn classifier.



(b) Testing data set 2 with 1-nn classifier.

(f) Comparing the outcomes from part (d) and (e):
Training error:

- Part (d) has zero training error
- Part (e) has zero training error

There is no difference on the training set. Testing error:

- Part (d) has testing error $\frac{5}{10}$
- Part (e) has zero training error

For this particular test set, 1-nn results in zero testing error, thus it is more suitable.

3. Robustness of the Majority Vote Classifier.

The purpose of this problem is to further investigate the behavior of the majority vote classifier (*see slides 5-7 of lecture 12*) using Hoeffding's inequality (*see slide 7 of lecture 12, and for more background, slide 14 of lecture 13*). Simplified versions of Hoeffding's inequality are as follows.

Theorem 1. *Given independent random variables* $(Z_1, \ldots, Z_k)$ *with* $Z_i \in [0,1]$,

$$\Pr\left[\sum_{i=1}^{k} Z_i \geq \sum_{i=1}^{k} \mathbb{E}[Z_i] + k\epsilon\right] \leq \exp\left(-2k\epsilon^2\right), \tag{1}$$

and

$$\Pr\left[\sum_{i=1}^{k} Z_i \leq \sum_{i=1}^{k} \mathbb{E}[Z_i] - k\epsilon\right] \leq \exp\left(-2k\epsilon^2\right). \tag{2}$$

In this problem we have an odd number n of classifiers $\{f_1, \ldots, f_n\}$ and only consider their behavior on a fixed data example $(\boldsymbol{x}, y)$; by classifier we mean $f_i(\boldsymbol{x}) \in \{\pm 1\}$. Define the majority vote classifer MAJ as

$$\textsc{Maj}(\boldsymbol{x}; f_1, \ldots, f_n) := 2 \cdot \mathbb{1}\left[\sum_{i=1}^{n} f_i(\boldsymbol{x}) \geq 0\right] - 1 = \begin{cases} +1 & \sum_{i=1}^{n} f_i(\boldsymbol{x}) > 0, \\ -1 & \sum_{i=1}^{n} f_i(\boldsymbol{x}) < 0, \end{cases}$$

where we will not need to worry about ties since n is odd.

To demonstrate the utility of Theorem 1 in analyzing MAJ, suppose that $\Pr[f_i(\boldsymbol{x}) = y] = p > 1/2$ independently for each i. Then, by defining a random variable $Z_i := \mathbb{1}[f_i(\boldsymbol{x}) \neq y]$ and noting $\mathbb{E}[Z_i] = 1 - p$,

$$\begin{aligned} \Pr[\textsc{Maj}(\boldsymbol{x}; f_1, \ldots, f_n) \neq y] &= \Pr\left[\sum_{i=1}^{n} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] \geq \frac{n}{2}\right] \\ &= \Pr\left[\sum_{i=1}^{n} Z_i \geq n(1-p) - \frac{n}{2} + np\right] \\ &= \Pr\left[\sum_{i=1}^{n} Z_i \geq n\mathbb{E}[Z_1] + n(p - 1/2)\right] \\ &\leq \exp\left(-2n(p-1/2)^2\right). \end{aligned}$$

The purpose of this problem is to study the behavior of $\textsc{Maj}(\boldsymbol{x})$ when not all of the classifiers $\{f_1, \ldots, f_n\}$ are independent.

(a) [hw3] Assume n is divisible by 7 and $5n/7$ is odd, and that of the n classifiers $\{f_1, \ldots, f_n\}$, now only the first $5n/7$ of them (i.e., $\{f_1, \ldots, f_{5n/7}\}$) have independent errors on $\boldsymbol{x}$. Specifically, $\Pr[f_i(\boldsymbol{x}) = y] = p := 4/5$ for classifiers $\{f_1, \ldots, f_{5n/7}\}$. By contrast, we make no assumption on the other $2n/7$ classifiers (i.e., $\{f_{5n/7+1}, \ldots, f_n\}$) and their errors. Now use Hoeffding's inequality to show that

$$\Pr\left[\sum_{i=1}^{5n/7} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] \geq \frac{3n}{14}\right] \leq \exp\left(-\frac{n}{70}\right).$$

(b) [hw3] Continuing from (a), further show that the majority vote classifier over all n classifiers is still good, specifically showing

$$\Pr\left[\textsc{Maj}(\boldsymbol{x}; f_1, \ldots, f_n) \neq y\right] \leq \exp\left(-\frac{n}{70}\right).$$

For full points: You need to derive the inequality $\Pr\left[\textsc{Maj}(\boldsymbol{x}; f_1, \ldots, f_n) \neq y\right] \leq \exp(-n/70)$ rigorously for ANY possible behavior of the $\frac{2n}{7}$ arbitrary classifiers.

(c) [hw3] Is the probability of correctly classifying $\boldsymbol{x}$ reasonably good in part (b) for large n? Do you have any interesting observations? Any answer which contains at least one complete sentence will receive full credit.

(d) [hw3] Now suppose that n is divisible by 5 and $3n/5$ is odd, but now only first $3n/5$ of the classifiers (i.e., $\{f_1, \ldots, f_{3n/5}\}$) have independent errors, and are correct with probability $\Pr[f_i(\boldsymbol{x}) = y] = p := 2/3$. Use Hoeffding's inequality to show that

$$\Pr\left[\sum_{i=1}^{3n/5} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] \leq \frac{n}{10}\right] \leq \exp\left(-\frac{n}{30}\right).$$

(e) [hw3] Continuing from (d), describe malicious behavior for the remaining $2n/5$ classifiers so that

$$\Pr\left[\textsc{Maj}(\boldsymbol{x}; f_1, \ldots, f_n) = y\right] \leq \exp\left(-\frac{n}{30}\right).$$

For full points: Describe the malicious behavior of the arbitrary classifiers AND derive the inequality $\Pr\left[\textsc{Maj}(\boldsymbol{x}; f_1, \ldots, f_n) = y\right] \leq \exp(-n/30)$.

(f) [hw3] Comparing the results from part (b) and part (e), do you have any observation? Any answer which contains at least one complete sentence will receive full credit.

Solution.

(a) According to the definition of Hoeffding's inequality:

$$\Pr\left[\sum_{i=1}^{k} Z_i \geq \sum_{i=1}^{k} \mathbb{E}[Z_i] + k\epsilon\right] \leq \exp\left(-2k\epsilon^2\right)$$

Replacing k with $\frac{5n}{7}$,

$$\Pr\left[\sum_{i=1}^{\frac{5n}{7}} Z_i \geq \sum_{i=1}^{\frac{5n}{7}} \mathbb{E}[Z_i] + \frac{5n}{7}\epsilon\right] \leq \exp\left(-2 * \frac{5n}{7}\epsilon^2\right)$$

$\sum_{i=1}^{\frac{5n}{7}} \mathbb{E}[Z_i]$ can be written as k*$(1 - \frac{4}{5})$. Thus:

$$\Pr\left[\sum_{i=1}^{\frac{5n}{7}} Z_i \geq \frac{5n}{7} * \frac{1}{5} + \frac{5n}{7}\epsilon\right] \tag{3}$$

The LHS from the question can be rewritten as:

$$\Pr\left[\sum_{i=1}^{5n/7} Z_i \geq \frac{3n}{14}\right] \tag{4}$$

(Based on the utility of Theorem 1 derivation).
Comparing (3) and (4):

$$\frac{3n}{14} = \frac{5n}{7}(\frac{1}{5} + \epsilon)$$

On solving the above equation for ϵ, the value received is

$$\epsilon = \frac{1}{10}$$

Substituting this ϵ value in the original definition equation RHS:

$$\exp\left(-2 * \frac{5n}{7}\epsilon^2\right) = \exp\left(-2 * \frac{5n}{7} * (\frac{1}{10})^2\right)$$
$$= \exp\left(-\frac{n}{70}\right)$$

Thus, equation in part (a) holds true.

(b) We know that:

$$Pr[(\boldsymbol{x}; f_1, \ldots, f_n) \neq y] = Pr\left[\sum_{i=1}^{n} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] \geq \frac{n}{2}\right]$$

For the given data, the RHS in the above equation can be rewritten as:

$$Pr\left[\sum_{i=1}^{\frac{5n}{7}} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] + \sum_{i=\frac{5n}{7}+1}^{n} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] \geq \frac{n}{2}\right]$$

The value of $\mathbb{1}[f_i(\boldsymbol{x}) \neq y]$ will be 1 when there is a mis-classification. For the $\frac{2n}{7}$ classifiers, if all of them are wrong, then the maximum value that $\sum_{i=\frac{5n}{7}+1}^{n} \mathbb{1}[f_i(\boldsymbol{x}) \neq y]$ can have is $\frac{2n}{7}$. Considering this worst case scenario in the above equation:

$$Pr\left[\sum_{i=1}^{\frac{5n}{7}} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] + \frac{2n}{7} \geq \frac{n}{2}\right] = Pr\left[\sum_{i=1}^{\frac{5n}{7}} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] \geq \frac{n}{2} - \frac{2n}{7}\right] = Pr\left[\sum_{i=1}^{\frac{5n}{7}} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] \geq \frac{3n}{14}\right]$$

From (a), we know that the above equation is upper bounded by:

$$Pr\left[\sum_{i=1}^{\frac{5n}{7}} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] \geq \frac{3n}{14}\right] \leq \exp(-n/70)$$

$$=> \Pr\left[(\boldsymbol{x}; f_1, \ldots, f_n) \neq y\right] \leq \exp\left(-\frac{n}{70}\right)$$

In case none of the classifiers are wrong, the outcome we get on RHS is:

$$=> \Pr\left[(\boldsymbol{x}; f_1, \ldots, f_n) \neq y\right] \leq \exp\left(-\frac{9n}{50}\right)$$

(from the utility of Theorem 1 derivation, substituting p = 4/5)
For any n¿0, the above equation can be written as:

$$=> \Pr\left[(\boldsymbol{x}; f_1, \ldots, f_n) \neq y\right] \leq \exp\left(-\frac{n}{70}\right)$$

Thus, (b) is proved.

(c) Yes.
As $n -> \infty$, based on part (a):

$$\Pr\left[\sum_{i=1}^{5n/7} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] \geq \frac{3n}{14}\right] \leq \exp\left(-\frac{n}{70}\right)$$

$$\exp\left(-\frac{\infty}{70}\right) = 0$$

Thus, according to the equation, the error will be zero.

(d) According to the definition of Hoeffding's inequality:

$$\Pr\left[\sum_{i=1}^{k} Z_i \leq \sum_{i=1}^{k} \mathbb{E}[Z_i] - k\epsilon\right] \leq \exp\left(-2k\epsilon^2\right)$$

Replacing k with $\frac{3n}{5}$,

$$\Pr\left[\sum_{i=1}^{\frac{3n}{5}} Z_i \leq \sum_{i=1}^{\frac{3n}{5}} \mathbb{E}[Z_i] - \frac{3n}{5}\epsilon\right] \leq \exp\left(-2 * \frac{3n}{5}\epsilon^2\right)$$

$\sum_{i=1}^{\frac{3n}{5}} \mathbb{E}[Z_i]$ can be written as k*$(1 - \frac{2}{3})$. Thus:

$$\Pr\left[\sum_{i=1}^{\frac{3n}{5}} Z_i \leq \frac{3n}{5} * \frac{1}{3} + \frac{3n}{5}\epsilon\right] \tag{5}$$

The LHS from the question can be rewritten as:

$$\Pr\left[\sum_{i=1}^{3n/5} Z_i \leq \frac{n}{10}\right] \tag{6}$$

(Based on the utility of Theorem 1 derivation).
Comparing (5) and (6):

$$\frac{n}{10} = \frac{3n}{5}(\frac{1}{3} - \epsilon)$$

On solving the above equation for ϵ, the value received is

$$\epsilon = \frac{1}{6}$$

Substituting this ϵ value in the original definition equation RHS:

$$\begin{aligned}\exp\left(-2 * \frac{3n}{5}\epsilon^2\right) &= \exp\left(-2 * \frac{3n}{5} * (\frac{1}{6})^2\right) \\ &= \exp\left(-\frac{n}{30}\right)\end{aligned}$$

Thus, equation in part (d) holds true.

(e) We know that:

$$Pr[(\boldsymbol{x}; f_1, \ldots, f_n) \neq y] = Pr\left[\sum_{i=1}^{n} \mathbb{1}[f_i(\boldsymbol{x}) \neq y] \geq \frac{n}{2}\right]$$

Thus,

$$Pr[(\boldsymbol{x}; f_1, \ldots, f_n) = y] = Pr\left[\sum_{i=1}^{n} \mathbb{1}[f_i(\boldsymbol{x}) = y] \leq \frac{n}{2}\right]$$

For the given data, the RHS in the above equation can be rewritten as:

$$Pr\left[\sum_{i=1}^{\frac{3n}{5}} \mathbb{1}[f_i(\boldsymbol{x}) = y] + \sum_{j=\frac{3n}{5}+1}^{n} \mathbb{1}[f_i(\boldsymbol{x}) = y] \leq \frac{n}{2}\right]$$

The value of $\mathbb{1}[f_i(\boldsymbol{x}) \neq y]$ will be 1 when there is a mis-classification. For the $\frac{2n}{5}$ classifiers, if all of them are wrong, then the maximum value that $\sum_{i=\frac{3n}{5}+1}^{n} \mathbb{1}[f_i(\boldsymbol{x}) \neq y]$ can have is $\frac{2n}{5}$. Considering this worst case scenario in the above equation:

$$Pr\left[\sum_{i=1}^{\frac{3n}{5}} \mathbb{1}[f_i(\boldsymbol{x}) = y] + \frac{2n}{5} \leq \frac{n}{2}\right] = Pr\left[\sum_{i=1}^{\frac{3n}{5}} \mathbb{1}[f_i(\boldsymbol{x}) = y] \leq \frac{n}{2} - \frac{2n}{5}\right] = Pr\left[\sum_{i=1}^{\frac{3n}{5}} \mathbb{1}[f_i(\boldsymbol{x}) = y] \leq \frac{n}{10}\right]$$

From (d), we know that the above equation is lower bounded by:

$$Pr\left[\sum_{i=1}^{\frac{3n}{5}} \mathbb{1}[f_i(\boldsymbol{x}) = y] \leq \frac{n}{10}\right] \leq \exp(-n/30)$$

$$=> \Pr\left[(\boldsymbol{x}; f_1, \ldots, f_n) = y\right] \leq \exp\left(-\frac{n}{30}\right)$$

For any n¿0, the above equation can be written as:

$$=> \Pr\left[(\boldsymbol{x}; f_1, \ldots, f_n) = y\right] \leq \exp\left(-\frac{n}{30}\right)$$

Thus, (e) is proved.

(f) The probability values calculated in (b) and (e) measure the opposite metric. While (b) gives probability of mis-classification, (e) gives the probability of correct classification. Considering the RHS of both the equations, purely on the basis of denominator, for n¿0, (e) seems to give higher values.

References

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.