Homework #3

CSE 446: Machine Learning
Prof. Sewoong Oh
Due: **Friday** February 25th, 2022 11:59pm
81 points

Please review all homework guidance posted on the website before submitting to GradeScope. Reminders:

- Make sure to read the "What to Submit" section following each question and include all items.
- Please provide succinct answers and supporting reasoning for each question. Similarly, when discussing experimental results, concisely create tables and/or figures when appropriate to organize the experimental results. All explanations, tables, and figures for any particular part of a question must be grouped together.
- For every problem involving generating plots, please include the plots as part of your PDF submission.
- When submitting to Gradescope, please link each question from the homework in Gradescope to the location of its answer in your homework PDF. Failure to do so may result in deductions of up to *[5 points]* . For instructions, see `https://www.gradescope.com/get_started#student-submission`.
- If you collaborate on this homework with others, you must indicate who you worked with on your homework. Failure to do so may result in accusations of plagiarism.
- Please indicate your final answer to each question by placing a box around the main result(s). To do this in LaTeX, one option is using the `boxed` command.
- For problems asking you to write some code, please submit code both to Gradescope and **embed the relevant code snippet in your PDF**. Failure to embed code in your PDF will result in point deductions.

Not adhering to these reminders may result in point deductions.

Note on Homework Length — This homework may take slightly longer than previous homeworks due to coding implementations (kernel ridge regression and neural networks). Plan accordingly: **please start early** (look over all problems so you can work efficiently), ask questions on Ed, come to office hours, and ask your TAs/instructor if you need any help!

Changelog:

- **2/22:** Fixed the footnote in page 3. $||x_i - x_j||_2$ is fixed to be $||x_i - x_j||_2^2$
- **2/22:** Fixed A2.c wording. The phrase "Show that the minimum squared distance to ..." is now changed to be "Show that the minimum distance to ..."

Conceptual Questions

A1. The answers to these questions should be answerable without referring to external materials. Briefly justify your answers with a few words.

a. *[2 points]* Say you trained an SVM classifier with an RBF kernel ($K(u,v) = \exp\left(-\frac{\|u-v\|_2^2}{2\sigma^2}\right)$). It seems to underfit the training set: should you increase or decrease σ?

b. *[2 points]* True or False: Training deep neural networks requires minimizing a non-convex loss function, and therefore gradient descent might not reach the globally-optimal solution.

c. *[2 points]* True or False: It is a good practice to initialize all weights to zero when training a deep neural network.

d. *[2 points]* True or False: We use non-linear activation functions in a neural network's hidden layers so that the network learns non-linear decision boundaries.

e. *[2 points]* True or False: Given a neural network, the time complexity of the backward pass step in the backpropagation algorithm can be prohibitively larger compared to the relatively low time complexity of the forward pass step.

f. *[2 points]* True or False: Neural Networks are the most extensible model and therefore the best choice for any circumstance.

What to Submit:

- **Parts a-f:** 1-2 sentence explanation containing your answer.

Support Vector Machines

A2. Assume w is an n-dimensional vector and b is a scalar. A hyperplane in $\mathbb{R}^n$ is the set $\left\{x \in \mathbb{R}^n \;\middle|\; w^\top x + b = 0\right\}$.

a. *[1 point]* ($n = 2$ example) Draw the hyperplane for $w = \begin{bmatrix}-1 & 2\end{bmatrix}^\top$, $b = 2$? Label your axes.

b. *[1 point]* ($n = 3$ example) Draw the hyperplane for $w = \begin{bmatrix}1 & 1 & 1\end{bmatrix}^\top$, $b = 0$? Label your axes.

c. *[2 points]* Let $w^\top x + b = 0$ be the hyperplane generated by a certain SVM optimization. Given some point $x_0 \in \mathbb{R}^n$, Show that the minimum *distance* to the hyperplane is $\frac{|x_0^\top w + b|}{\|w\|_2}$. In other words, show the following:

$$\min_x \|x_0 - x\|_2 = \frac{|x_0^\top w + b|}{\|w\|_2}$$
$$\text{subject to: } w^\top x + b = 0\,.$$

Hint: Think about projecting x_0 onto the hyperplane.

What to Submit:

- **Parts a-b:** Graph or image of hyperplane
- **Part c:** Solution and corresponding calculations

Kernels and Bootstrap

A3. *[5 points]* Suppose that our inputs x are one-dimensional and that our feature map is infinite-dimensional: $\phi(x)$ is a vector whose ith component is:

$$\frac{1}{\sqrt{i!}} e^{-x^2/2} x^i \ ,$$

for all nonnegative integers i. (Thus, ϕ is an infinite-dimensional vector.) Show that $K(x, x') = e^{-\frac{(x-x')^2}{2}}$ is a kernel function for this feature map, i.e.,

$$\phi(x) \cdot \phi(x') = e^{-\frac{(x-x')^2}{2}} \ .$$

Hint: Use the Taylor expansion of $z \mapsto e^z$. (This is the one dimensional version of the Gaussian (RBF) kernel).

What to Submit:

- Proof.

A4. This problem will get you familiar with kernel ridge regression using the polynomial and RBF kernels. First, let's generate some data. Let $n = 30$ and $f_*(x) = 4\sin(\pi x)\cos(6\pi x^2)$. For $i = 1, \ldots, n$ let each x_i be drawn uniformly at random from $[0, 1]$, and let $y_i = f_*(x_i) + \epsilon_i$ where $\epsilon_i \sim \mathcal{N}(0, 1)$. For any function f, the true error and the train error are respectively defined as:

$$\mathcal{E}_{\text{true}}(f) = \mathbb{E}_{X,Y}\left[(f(X) - Y)^2\right], \qquad \widehat{\mathcal{E}}_{\text{train}}(f) = \frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y_i)^2 \ .$$

Now, our goal is, using kernel ridge regression, to construct a predictor:

$$\widehat{\alpha} = \arg\min_{\alpha} \|K\alpha - y\|_2^2 + \lambda \alpha^\top K \alpha \ , \qquad \widehat{f}(x) = \sum_{i=1}^{n} \widehat{\alpha}_i k(x_i, x)$$

where $K \in \mathbb{R}^{n \times n}$ is the kernel matrix such that $K_{i,j} = k(x_i, x_j)$, and $\lambda \geq 0$ is the regularization constant.

a. *[10 points]* Using leave-one-out cross validation, find a good λ and hyperparameter settings for the following kernels:

- $k_{\text{poly}}(x, z) = (1 + x^\top z)^d$ where $d \in \mathbb{N}$ is a hyperparameter,
- $k_{\text{rbf}}(x, z) = \exp(-\gamma\|x - z\|_2^2)$ where $\gamma > 0$ is a hyperparameter[1].

We strongly recommend implementing either grid search or random search. **Do not use sklearn**, but actually implement of these algorithms. Reasonable values to look through in this problem are: $\lambda \in 10^{[-5,-1]}$, $d \in [5, 25]$, γ sampled from a narrow gaussian distribution centered at value described in the footnote.
Report the values of d, γ, and the λ values for both kernels.

b. *[10 points]* Let $\widehat{f}_{\text{poly}}(x)$ and $\widehat{f}_{\text{rbf}}(x)$ be the functions learned using the hyperparameters you found in part a. For a single plot per function $\widehat{f} \in \left\{\widehat{f}_{\text{poly}}(x), \widehat{f}_{\text{rbf}}(x)\right\}$, plot the original data $\{(x_i, y_i)\}_{i=1}^n$, the true $f(x)$, and $\widehat{f}(x)$ (i.e., define a fine grid on $[0, 1]$ to plot the functions).

c. *[5 points]* We wish to build bootstrap percentile confidence intervals for $\widehat{f}_{\text{poly}}(x)$ and $\widehat{f}_{\text{rbf}}(x)$ for all $x \in [0, 1]$ from part b.[2] Use the non-parametric bootstrap with $B = 300$ bootstrap iterations to find 5% and 95% percentiles at each point x on a fine grid over $[0, 1]$.

[1] Given a dataset $x_1, \ldots, x_n \in \mathbb{R}^d$, a heuristic for choosing a range of γ in the right ballpark is the inverse of the median of all $\binom{n}{2}$ squared distances $\|x_i - x_j\|_2^2$.

[2] See Hastie, Tibshirani, Friedman Ch. 8.2 for a review of the bootstrap procedure.

Specifically, for each bootstrap sample $b \in \{1, \ldots, B\}$, draw uniformly at randomly with replacement, n samples from $\{(x_i, y_i)\}_{i=1}^n$, train an $\widehat{f}_b$ using the b-th resampled dataset, compute $\widehat{f}_b(x)$ for each x in your fine grid; let the 5% percentile at point x be the largest value ν such that $\frac{1}{B}\sum_{b=1}^{B} \mathbf{1}\{\widehat{f}_b(x) \leq \nu\} \leq .05$, define the 95% percentile analogously.

Plot the 5 and 95 percentile curves on the plots from part b.

d. *[5 points]* Repeat parts a, b, and c with $n = 300$, but use 10-fold CV instead of leave-one-out for part a.

e. *[5 points]* For this problem, use the $\widehat{f}_{\text{poly}}(x)$ and $\widehat{f}_{\text{rbf}}(x)$ learned in part d. Suppose $m = 1000$ additional samples $(x'_1, y'_1), \ldots, (x'_m, y'_m)$ are drawn i.i.d. the same way the first n samples were drawn.

Use the non-parametric bootstrap with $B = 300$ to construct a confidence interval on

$$\mathbb{E}\left[\left(Y - \widehat{f}_{\text{poly}}(X)\right)^2 - \left(Y - \widehat{f}_{\text{rbf}}(X)\right)^2\right]$$

(i.e. randomly draw with replacement m samples denoted as $\{(\widetilde{x}'_i, \widetilde{y}'_i)\}_{i=1}^m$ from $\{(x'_i, y'_i)\}_{i=1}^m$ and compute $\frac{1}{m}\sum_{i=1}^{m}\left[\left(\widetilde{y}'_i - \widehat{f}_{\text{poly}}(\widetilde{x}'_i)\right)^2 - \left(\widetilde{y}'_i - \widehat{f}_{\text{rbf}}(\widetilde{x}'_i)\right)^2\right]$, repeat this B times) and find 5% and 95% percentiles. Report these values.

Using this confidence interval, is there statistically significant evidence to suggest that one of $\widehat{f}_{\text{rbf}}$ and $\widehat{f}_{\text{poly}}$ is better than the other at predicting Y from X? (Hint: does the confidence interval contain 0?)

What to Submit:

- **Part a:** Report the values of d, γ and the value of λ for both kernels as described.
- **Part b:** Two plots. One plot for each function.
- **Part c:** Two plots. One plot for the 5 and 95 percentile curves of each plot in part b.
- **Part d:** Values of d, γ, and the value of λ for both kernels as described. In addition, provide two separate plots as you did for part b. then two more plots as you did for part c.
- **Part e:** Report the 5% and 95% percentiles. In addition, in 1-2 sentences, comment on if there is statistically significant evidence to suggest that one kernel performs better than the other.
- **Code** on Gradescope through coding submission and also **embed your code in the PDF submission**.

Introduction to PyTorch

A5. PyTorch is a great tool for developing, deploying and researching neural networks and other gradient-based algorithms. In this problem we will explore how this package is built, and re-implement some of its core components. Firstly start by reading `README.md` file provided in `intro_pytorch` subfolder. A lot of problem statements will overlap between here, readme's and comments in functions.

a. *[10 points]* You will start by implementing components of our own PyTorch modules. You can find these in folders: `layers`, `losses` and `optimizers`. Almost each file there should contain at least one problem function, including exact directions for what to achieve in this problem. Lastly, you should implement functions in `train.py` file.

b. *[5 points]* Next we will use the above module to perform hyperparameter search. Here we will also treat loss function as a hyper-parameter. However, because cross-entropy and MSE require different shapes we are going to use two different files: `crossentropy_search.py` and `mean_squared_error_search.py`. For each you will need to build and train (in provided order) 5 models:

- Linear neural network (Single layer, no activation function)

- NN with one hidden layer (2 units) and sigmoid activation function after the hidden layer
- NN with one hidden layer (2 units) and ReLU activation function after the hidden layer
- NN with two hidden layer (each with 2 units) and Sigmoid, ReLU activation functions after first and second hidden layers, respectively
- NN with two hidden layer (each with 2 units) and ReLU, Sigmoid activation functions after first and second hidden layers, respectively

For each loss function, submit a plot of losses from training and validation sets. All models should be on the same plot (10 lines per plot), with two plots total (1 for MSE, 1 for cross-entropy).

c. *[5 points]* For each loss function, report the best performing architecture (best performing is defined here as achieving the lowest validation loss at any point during the training), and plot its guesses on test set. You should use function `plot_model_guesses` from `train.py` file. Lastly, report accuracy of that model on a test set.

d. *[3 points]* Is there a big gap in performance between between MSE and Cross-Entropy models? If so, explain why it occurred? If not explain why different loss functions achieve similar performance? Answer in 2-4 sentences.

On Softmax function

One of the activation functions we ask you to implement is softmax. For a prediction $\hat{y} \in \mathbb{R}^k$ corresponding to single datapoint (in a problem with k classes):

$$\text{softmax}(\hat{y}_i) = \frac{\exp(\hat{y}_i)}{\sum_j \exp(\hat{y}_j)}$$

What to Submit:

- **Part b:** 2 plots (one per loss function), with 10 lines each, showing both training and validation loss of each model. Make sure plots are titled, and have proper legends.
- **Part c:** Names of best performing models (i.e. descriptions of their architectures), and their accuracy on test set.
- **Part c:** 2 scatter plots (one per loss function), with predictions of best performing models on test set.
- **Part d:** 2-4 sentence written reponse to provided questions.
- **Code** on Gradescope through coding submission and also **embed your code in the PDF submission**.

Administrative

A6.

a. *[2 points]* About how many hours did you spend on this homework? There is no right or wrong answer :)