XCS229ii Problem Set 1 — ML Theory

Due Thursday, 25 March at 11:59pm PT.

Guidelines

1. If you have a question about this homework, we encourage you to post your question on our Slack channel, at http://xcs229ii-scpd.slack.com/
2. Familiarize yourself with the collaboration and honor code policy before starting work.
3. For the coding problems, you must use the packages specified in the provided environment description. Since the autograder uses this environment, we will not be able to grade any submissions which import unexpected libraries.

Submission Instructions

Coding Submission: Some questions in this assignment require a coding response. For these questions, you should submit only the `src/submission.py` file in the online student portal. For further details, see Writing Code and Running the Autograder below.

Honor code

We strongly encourage students to form study groups. Students may discuss and work on homework problems in groups. However, each student must write down the solutions independently, and without referring to written notes from the joint session. In other words, each student must understand the solution well enough in order to reconstruct it by him/herself. In addition, each student should write on the problem set the set of people with whom s/he collaborated. Further, because we occasionally reuse problem set questions from previous years, we expect students not to copy, refer to, or look at the solutions in preparing their answers. It is an honor code violation to intentionally refer to a previous year's solutions. More information regarding the Stanford honor code can be foudn at https://communitystandards.stanford.edu/policies-and-guidance/honor-code.

Writing Code and Running the Autograder

All your code should be entered into `src/submission.py`. When editing `src/submission.py`, please only make changes between the lines containing `### START_CODE_HERE ###` and `### END_CODE_HERE ###`. Do not make changes to files other than `src/submission.py`.

The unit tests in `src/grader.py` (the autograder) will be used to verify a correct submission. Run the autograder locally using the following terminal command within the `src/` subdirectory:

```
$ python grader.py
```

There are two types of unit tests used by the autograder:

- `basic`: These tests are provided to make sure that your inputs and outputs are on the right track, and that the hidden evaluation tests will be able to execute.
- `hidden`: These unit tests are the evaluated elements of the assignment, and run your code with more complex inputs and corner cases. Just because your code passed the basic local tests does not necessarily mean that they will pass all of the hidden tests. These evaluative hidden tests will be run when you submit your code to the Gradescope autograder via the online student portal, and will provide feedback on how many points you have earned.

For debugging purposes, you can run a single unit test locally. For example, you can run the test case `3a-0-basic` using the following terminal command within the `src/` subdirectory:

```
$ python grader.py 3a-0-basic
```

Before beginning this course, please walk through the Anaconda Setup for XCS Courses to familiarize yourself with the coding environment. Use the env defined in `src/environment.yml` to run your code. This is the same environment used by the online autograder.

Introduction

This homework is a series of multiple choice questions focusing on Machine Learning theory. It includes a short, fictitious case study, some statistical theory revolving around the bias-variance tradeoff, and even a look into interesting new Machine Learning research!

How to submit: This assignment includes only multiple choice questions. Even though these are not coding questions, you will submit your response to each question in the `submission.py` file. The response to the first question looks like this:

```
def multiple_choice_1a():
  """"""
  # Return a python collection with the option(s) that you believe are correct
  # like this:
  # `return ['a']`
  # or
  # `return ['a', 'd']`
  response = []
  ### START CODE HERE ###
  ### END CODE HERE ###
  return response
```

If you believe that `a` and `b` are the correct responses to this question, you will type `response = ['a', 'b']` between the indicated lines like this:

```
def multiple_choice_1a():
  """"""
  # Return a python collection with the option(s) that you believe are correct
  # like this:
  # `return ['a']`
  # or
  # `return ['a', 'd']`
  response = []
  ### START CODE HERE ###
  response = ['a', 'b']
  ### END CODE HERE ###
  return response
```

How to verify your submission: You can run the student version of the autograder locally like all coding problem sets. In the case of this problem set, the helper tests will verify that your responses are within the set of possible choices for each question (e.g. the helper functions will flag if you forget to answer a question of if you respond with `['a', 'd']` when the choices are `['a', 'b', 'c']`.) See the previous page for instructions to run the autograder.

1. **City Council: Vehicle Detection Model**

This example is adapted from a real production application, but with details disguised to protect confidentiality.

The City Council wants to deploy a model that will detect whether or not there's a motorized vehicle in a given street. Your goal is to build an algorithm that will detect whether or not a motorized vehicle is in an image taken by a street's security camera. The City Council gives you a dataset of 1,000,000 labelled images taken by the streets security cameras. They are labelled:

- y=0: There is no motorized vehicle in the image
- y=1: There is a motorized vehicle in the image

There are a lot of decisions to make which includes but is not limited to defining the evaluation metric, deciding how to structure the data into train/val/test sets, etc. The goal of this problem is to give you the opportunity to apply concepts from machine learning theory to make important decisions on how to create an effective car detection model.

(a) **[1 point] Metric of Success.** The City Council tells you that they want an algorithm that

- Has high accuracy
- Runs quickly and takes only a short time to classify a new image
- Can fit in a small amount of memory, so that it can run in a small processor that the city will attach to security cameras

What are the pros and cons of having three different evaluation metrics? Select all that apply:

(a) Comparing two different algorithms becomes ambiguous. It's not inherently clear when one is better than another.

(b) Having to optimize three different things will likely result in more iterations for your team.

(c) In this case, having multiple evaluation metrics allows us to make sure we come up with a practical, useful algorithm.

(b) **[1 point] Memory vs Accuracy vs Speed.** Suppose the predictions are not going to be used in real time. Which ordering, from most to least important, best encapsulates the relative importance of accuracy, speed, and memory?

(a) Accuracy, Speed, Memory

(b) Speed, Accuracy, Memory

(c) Memory, Accuracy, Speed

(d) Speed, Memory, Accuracy

(c) **[1 point] Train-Dev-Test Splits.** Before implementing your algorithm, you need to split your data into train-dev-test sets. Which of these do you think is the best choice? Select one option below.

(a) Train: 200,000; Dev: 400,000; Test 400,000

(b) Train: 700,000; Dev: 250,000; Test 50,000

(c) Train: 600,000; Dev: 100,000; Test 300,000

(d) Train: 333,334; Dev: 333,333; Test 333,333

(d) **[2 points] Addding Data.** After setting up your train-dev-test sets, the City Council comes across another 1,000,000 images, called the "citizens' data". Apparently the citizens of the town took their own pictures from the sidewalk and labelled them, contributing these additional 1,000,000 images. These images are different from the distribution of images the City Council had originally given you (they were not images from security cameras), but you think it could help your algorithm. Select all that apply about using these images:

(a) You should not add this data to the training data because it comes from a different distribution than the dev and test data so it will hurt dev and test performance.

(b) You should not add this data to the dev set because then the dev set will no longer reflect the test set.

(c) You should not add this data to the test set because then the test set won't reflect the incoming data the model will be applied on.

(d) You should not add this data to the test set because a bigger test set will slow down the speed of iterating due to the computational expense of evaluating models on the test set.

(e) **[1 point] Error Analysis.** You train a system, and its errors are as follows (Error = 100% - Accuracy):

i. Training Error: 4.0%

ii. Dev Error: 4.5%

Team member one suggests training a bigger network so as to drive down the 4.0% Team member two claims you shouldn't increase the network's size because it is overfitting. Who do you agree with? Select one option below.

(a) Team member one because the 4.0% training error indicates high bias.

(b) Team member one because training a larger model will allow for better training and generalization error.

(c) Team member two because the dev error is significantly higher than the training error.

(d) There's not enough information to tell.

(f) **[1 point] Bayes Error.** Bayes error is the lowest possible error rate for any classifier of a random outcome (into, for example, one of two categories) and is analogous to the irreducible error (you can not reduce it anymore). Human-level performance is the best outcome for any human being. Which of the following statements do you agree with? Select one option below.

(a) A learning algorithm's performance can be better than human-level performance but it can never be better than Bayes error.

(b) A learning algorithm's performance can never be better than human-level performance but it can be better than Bayes error.

(c) A learning algorithm's performance can never be better than human-level performance nor better than Bayes error.

(d) A learning algorithm's performance can be better than human-level performance and better than Bayes error.

(g) **[1 point] Evaluate and Improve from Training-Dev Set.** Suppose you see the following human-level performance and training/dev performance

i. Human-level performance: 0.1%

ii. Training-set error: 2.0%

iii. Dev-set error: 1.9%

Based on the evidence you have, which two of the following five options seem the most promising to try? Select two options.

(a) Train a more complex model to try to do better on the training set.

(b) Train a less complex model on the training set to try and avoid overfitting.

(c) Get a bigger training set to reduce variance.

(d) Try increasing regularization.

(e) Try decreasing regularization.

(h) **[2 points] Evaluate and Improve from Test Set.** You also evaluate your model on the test set, and find the following:

i. Human-level performance: 0.1%

ii. Training-set performance: 2.0%

iii. Dev-set error: 2.1%

iv. Test-set error: 7.0%

What does this mean? Select the two best options.

(a) You overfit the dev set.

(b) You should try to get a bigger dev set to try and avoid overfitting.

(c) You have underfit the dev set.

(d) You should get a bigger test set

(i) **[1 point] Final Evaluation.** You also evaluate your model on the test set, and find the following:

i. Human-level performance: 0.1%

ii. Training-set performance: 0.05%

iii. Dev-set error: 0.05%

What can you conclude? Select all that apply.

(a) This is a statistical anomaly (or must be the result of statistical noise) since it should not be possible to surpass human-level performance.

(b) It's now harder to gauge avoidable bias, thus progress will be slower moving forward.

(c) With only 0.05% further progress to make, you should be able to close the gap to 0.00%

(d) If the dev set is big enough for the 0.05% error estimate to be accurate, then we know that the Bayes error <= 0.05%

(j) **[2 points] Dataset Size.** Suppose instead of giving you 1,000,000 images the city gave you 10,000,000 images, resulting in proportionally larger training, test, and dev sets. Which of the following statements do you agree with? Select all that apply.

(a) The increase in training set size will decrease the speed within which your team can iterate.

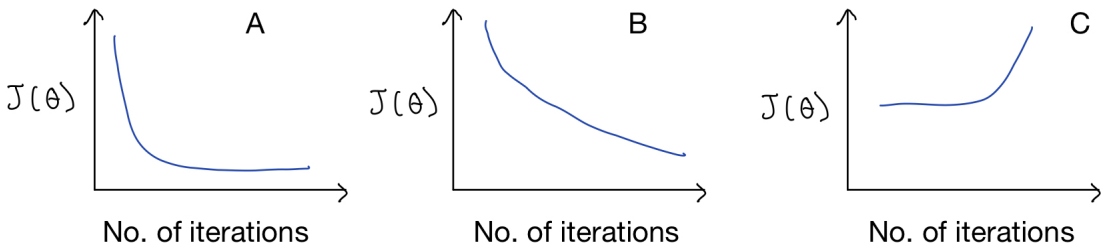
(b) Buying faster computers could speed up your team's iteration speed and thus increase productivity.

(c) Training and validating on a larger set of accurate data will likely result in a better model.

(d) It may make sense to only use a subset of k samples from the data in the case that the marginal performance benefit of using additional samples is small compared to the slow-down in iteration speed.

2. **Applying Machine Learning Theory**
Below are more example engaging with the application of machine learning theory. Enjoy!

(a) **[2 points] Graphical Analysis.**



Suppose the above graphs correspond to training the same model on the same data with stochastic gradient descent. Let l_1, l_2, and l_3 be the three learning rates for A, B, and C respectively. Which of the following is true about l_1, l_2, and l_3? Select one option below.

(a) $l_2 < l_1 < l_3$

(b) $l_1 > l_2 > l_3$

(c) $l_1 = l_2 = l_3$

(d) None of these

(b) **[1 point] Classification Model.** You are asked to build a classification model that predicts whether or not there will be an earthquake in California on a given day (a rare event). You build a model which achieves 98% accuracy on daily, incoming data for the next few months. Should you be confident that the model has learned something?

(a) Yes

(b) No

(c) **[1 point] Test vs Dev.** You're tuning hyperparameters for a model using cross-validation. After selecting your hyper parameters you try out the model on a test set. The test set performance is much worse than the dev performance. Which of the following may be the case? Select all that apply.

(a) You overfit the dev set.

(b) The test set distribution does not mirror the dev set distribution.

(c) You don't have enough dev data, so the dev error estimate is noisy.

(d) You're using too much regularization.

(d) **[2 points] Neural Networks.** Suppose you initialize all the weights in your neural network to 0. What will happen during training? Select one option below.

(a) The gradients will all be 0 so the weights will remain 0 throughout training.

(b) The neural net will never break symmetry (the weights across a particular hidden layer will remain identical) throughout training.

(c) The neural net will train normally and learn to fit the data.

(e) **[2 points] Non-linear vs Linear.** In the following four situations state whether you would rather use a non-linear class of models or linear class of models while training.

i. The sample size n is very large and the number of predictors p is very small.

(a) Linear

(b) Non-linear

ii. The number of predictors p is very large and the sample size n is very small.

(a) Linear

(b) Non-linear

iii. The relationship between the predictors and the response is highly polynomial.

(a) Linear

(b) Non-linear

iv. The variance of the errors is very high.

(a) Linear

(b) Non-linear

3. Double Descent

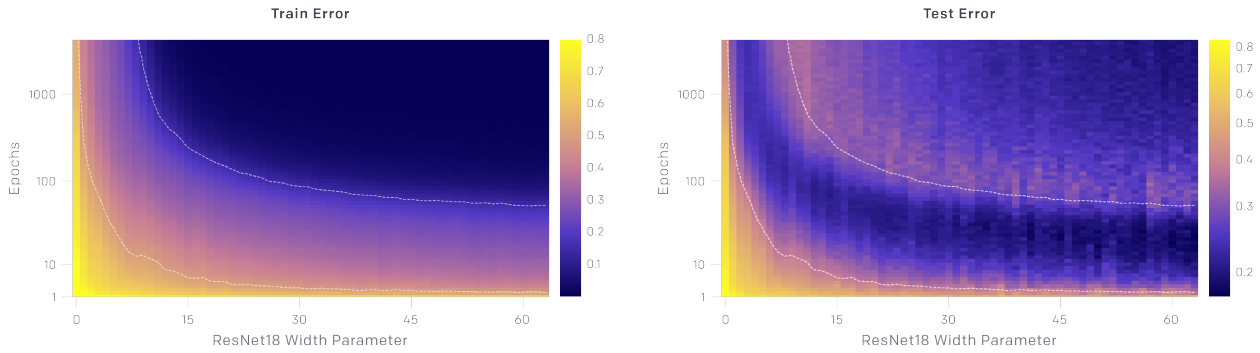
The following set of questions are going to discuss a phenomenon known as deep double descent. This is a modern research topic that is still not very well understood. Read the passage below and answer the related questions. For those interested in learning more, visit OpenAI's blog post here: https://openai.com/blog/deep-double-descent/

When selecting a class of models for a machine learning problem, a classical statistician would caution that using a large class will result in high variance/overfitting. A more contemporary ML practitioner may suggest using larger and larger models based on empirical findings over the last ten years. Recent work has shown that neither of these methodologies uphold exactly. Among many kinds of deep learning models we observe a double descent phenomenon where as model size/complexity increases, test performance first improves, then gets worse as the model begins to overfit, and then continues to improve as we increase model size/complexity. This peak in test error occurs in a 'critical regime', where the model is large/complex enough to almost interpolate (perfectly fit) the training set.

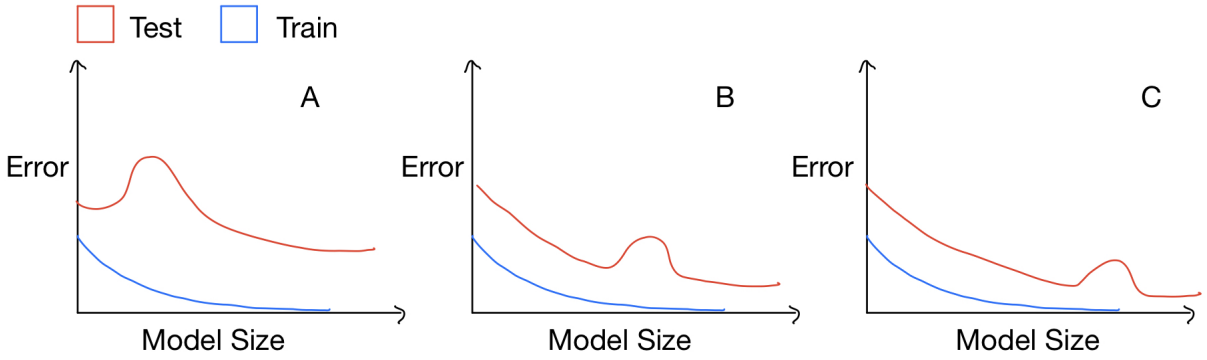
Intuitively, as one increases the size of an under-parameterized model that is unable to perfectly fit the training data, there is one specific weight configuration that best fits the data. Thus in the complexity regime where the model can approximately interpolate the training data, there is one weight configuration that results in this approximate interpolation. This weight configuration is unlikely to generalize to test data, hence the peak in test error. Once we enter the over-parameterized region there are many models which can perfectly fit the training data. For reasons we don't yet understand, the implicit bias of stochastic gradient descent (SGD) leads us to find models which both interpolate the training data and generalize well. Note that the implicit bias of SGD refers to the tendency over-parameterized models which are trained using SGD to generalize well to unseen data.

We also note a model size regime where having more training data hurts the performance of the model. This is illustrated in the plot below

Lastly, we note a phenomenon dubbed epoch-wise double descent, where the test error decreases with additional training, then increases as the training data is almost perfectly fit, and then continues to decrease. This is shown in the plot below:



(a) **[1 point] Plot Analysis.**



Which of the following plots of test/train accuracy would you most likely see when studying double descent? Select one option below.

(a) Plot A

(b) Plot B

(c) Plot C

(b) **[2 points] Explaining Double Descent.** Which of the following reasons explain why we see double descent? Select all that apply.

(a) The weights which allow an under-parameterized model to (over)fit the training data are unlikely to generalize to unseen data.

(b) Under-parameterized models are unable to fit the training data, hence they perform worse than over parameterized models.

(c) When models are overparameterized the randomness of SGD sometimes ends up training models which generalize well.

(d) In the over-parameterized regime the implicit bias of SGD results in finding solutions that generalize well to unseen data.

(c) **[1 point] Training Data.** Which of the following is the most likely reason for the regime where having more training data hurt the performance of the model? Select one option below.

(a) The additional data may not have the same distribution as the original data, making it harder for the model to find patterns in the data.

(b) It takes longer to train on more data, meaning there likely weren't as many iterations for tuning the hyperparameters.

(c) The size and location of the double descent peaks are unpredictable and not yet understood. This is one possible way they can manifest in the training/testing process.

(d) Since the critical regime for training on more data is shifted to the right, the double descent peaks don't line up. This results in the behavior we see.

4. **Bias-Variance Tradeoff**

Let's formally derive the bias-variance trade-off under some mild assumptions. For the sake of our treatment we will assume that we have a fixed set of x values which are non-random. For those more familiar with statistics, the same analysis can be done when x is random by conditioning on x. We will assume that for each of our samples x_i there is a corresponding random response $y_i = f(x_i) + \epsilon_i$ where f is a deterministic function and the ϵ_i are independent and identically distributed with mean $\mathbf{E}[\epsilon_i] = 0$ and some variance $\text{Var}[\epsilon_i] = \sigma^2$. Suppose we use some data set $\mathcal{D} := \{(x_i, y_i)\}$ to come up with (train) a predictive function $\hat{f}(x; \mathcal{D})$. The set $\mathcal{D}$ is our training data. We are interested in the average square error of our predictive function on out of sample data. Namely if (x, y) is a sample not in $\mathcal{D}$, then we want to understand $\mathbf{E}[(y - \hat{f}(x; \mathcal{D}))^2]$. Note in practice we use test error as an empirical estimate for this value.

Using the setup above, we can derive

$$
\begin{aligned}
&\mathbf{E}\left[\left(y - \hat{f}(x;\mathcal{D})\right)^2\right] \\
&= \mathbf{E}\left[\left(f(x) + \epsilon - \hat{f}(x;\mathcal{D})\right)^2\right] \\
&= \mathbf{E}\left[\left(f(x) - \mathbf{E}[\hat{f}(x;\mathcal{D})] + \epsilon - (\hat{f}(x;\mathcal{D}) - \mathbf{E}[\hat{f}(x;\mathcal{D})])\right)^2\right] \\
&= \mathbf{E}\left[\left(f(x) - \mathbf{E}[\hat{f}(x;\mathcal{D})]\right)^2\right] + \mathbf{E}\left[\epsilon^2\right] + \mathbf{E}\left[\left(\hat{f}(x;\mathcal{D}) - \mathbf{E}[\hat{f}(x;\mathcal{D})]\right)^2\right] + \cancelto{0}{\text{cross terms}} \\
&= \left(f(x) - \mathbf{E}[\hat{f}(x;\mathcal{D})]\right)^2 + \mathbf{E}\left[\epsilon^2\right] + \mathbf{E}\left[\left(\hat{f}(x;\mathcal{D}) - \mathbf{E}[\hat{f}(x;\mathcal{D})]\right)^2\right] \\
&= \text{Bias}^2[\hat{f}(x;\mathcal{D})] + \text{Var}(\hat{f}(x;\mathcal{D})) + \sigma^2
\end{aligned}
$$

where we've used the fact that $\text{Bias}^2[\hat{f}(x;\mathcal{D})] := (f(x) - \mathbf{E}[\hat{f}(x;\mathcal{D})])^2$ We leave it to you to confirm that the cross terms are indeed zero.

Using the above derivation, answer the following questions:

Select all the statements which are true. Suppose we find our predictive model by selecting the function $\hat{f}$ from a class of functions $\mathcal{F}$ which has the least cumulative squared error on the training data $\mathcal{D}$.

(a) **[1 point]** $\mathcal{D}$ **Size.** Increasing the size of $\mathcal{D}$ will likely (select all that apply):

- (a) Decrease the out of sample mean squared error because $\hat{f}$ will better estimate f, thereby decreasing the squared bias
- (b) Increase the out of sample mean squared error because $\hat{f}$ will worse estimate f, thereby increasing the squared bias
- (c) Decrease the out of sample mean squared error because $\hat{f}$ will be less variable, thereby decreasing the variance
- (d) Increase the out of sample mean squared error because $\hat{f}$ will be more variable, thereby increasing the variance

(b) **[2 points]** $\mathcal{F}$ **Increase.** Increasing the size of $\mathcal{F}$ will likely (select all that apply):

- (a) Decrease the out of sample mean squared error because $\hat{f}$ will better estimate f, thereby decreasing the squared bias
- (b) Increase the out of sample mean squared error because $\hat{f}$ will worse estimate f, thereby increasing the squared bias

(c) Decrease the out of sample mean squared error because $\hat{f}$ will be less variable, thereby decreasing the variance

(d) Increase the out of sample mean squared error because $\hat{f}$ will be more variable, thereby increasing the variance

(c) **[2 points]** $\mathcal{F}$ **Decrease.** Decreasing the size of $\mathcal{F}$ will likely (select all that apply):

(a) Decrease the out of sample mean squared error because $\hat{f}$ will better estimate f, thereby decreasing the squared bias

(b) Increase the out of sample mean squared error because $\hat{f}$ will worse estimate f, thereby increasing the squared bias

(c) Decrease the out of sample mean squared error because $\hat{f}$ will be less variable, thereby decreasing the variance

(d) Increase the out of sample mean squared error because $\hat{f}$ will be more variable, thereby increasing the variance