CS 189 Introduction to Machine Learning
Fall 2017 HW1

This homework is due **Friday, September 1 at 12pm noon.**

1 Getting Started

You may typeset your homework in latex or submit neatly handwritten and scanned solutions. Please make sure to start each question on a new page, as grading (with Gradescope) is much easier that way! Deliverables:

1. Submit a PDF of your writeup to assignment on Gradescope, "HW1 Write-Up"
2. Submit all code needed to reproduce your results, "HW1 Code".
3. Submit your test set evaluation results, "HW1 Test Set".

After you've submitted your homework, be sure to watch out for the self-grade form.

(a) Before you start your homework, write down your team. Who else did you work with on this homework? List names and email addresses. In case of course events, just describe the group. How did you work on this homework? Any comments about the homework?

(b) Please copy the following statement and sign next to it:

I certify that all solutions are entirely in my words and that I have not looked at another student's solutions. I have credited all external sources in this write up.

2 “Sample Complexity” of coupon collecting

Roald Dahl’s mathematical version of his book has the following problem. Willy Wonka has hidden 50 different types of cards in his chocolate wrappers, and some of the people to find all 50 types will be allowed to enter his factory and participate in a questionable experiment. Charlie goes to a local store to buy his chocolates, and can afford at most one chocolate a day. The store stocks 20 chocolates of each card type, and every time a chocolate is bought, it is replaced with another chocolate containing an identical card. Charlie picks a chocolate uniformly at random.

(a) What is the expected number of days it takes Charlie to qualify for Willy Wonka’s challenge if he picks a random chocolate every time he goes to the shop?

(b) The game has now changed. Instead of having to collect all d card types, at the end of the game Willy Wonka will pick a card uniformly at random and if someone has that card in their collection, they win a smaller prize. **If you want your probability of winning to be at least $1-\delta_w$, how many cards should you have in your collection?**

(c) Charlie just decides to buy chocolates for n days. At the end of this, he has some random number of distinct cards. **Given that there are d distinct cards and assuming that Charlie buys αd chocolates, what is the probability that Charlie wins a prize?**

(d) What happens as d gets large?

(e) Now, think about something that might seem completely different. We want to learn a completely unstructured function f on a finite domain of size d. We get training data from random samples, each of which picks a domain point U uniformly at random and then returns us $(U, f(U))$ — i.e. the domain point along with the function f’s evaluation on that domain point. **How big of a training set should we collect so that with probability $1-\delta$ we will be able to successfully estimate the function when presented with a uniformly drawn domain point.**

3 The accuracy of learning decision boundaries

This problem exercises your basic probability (e.g. from 70) in the context of understanding why lots of training data helps improve the accuracy of learning things.

For each $\theta \in (1/4, 3/4)$, define $f_\theta : [0,1] \to \{0,1\}$, such that

$$f_\theta(x) = \begin{cases} 1 \text{ if } x > \theta \\ 0 \text{ otherwise.} \end{cases}$$

The function is plotted in Figure **??**.

We draw samples $X_1, X_2, \ldots, X_n$ uniformly at random and i.i.d. from the interval $[0,1]$. Our goal is to learn an estimate for θ from n random samples $(X_1, f_\theta(X_1)), (X_2, f_\theta(X_2)), \ldots, (X_n, f_\theta(X_n))$.

Let $N_0 := \#\{i : f_\theta(X_i) = 0\}$ denote the number of samples we obtain that have function evaluation 0.

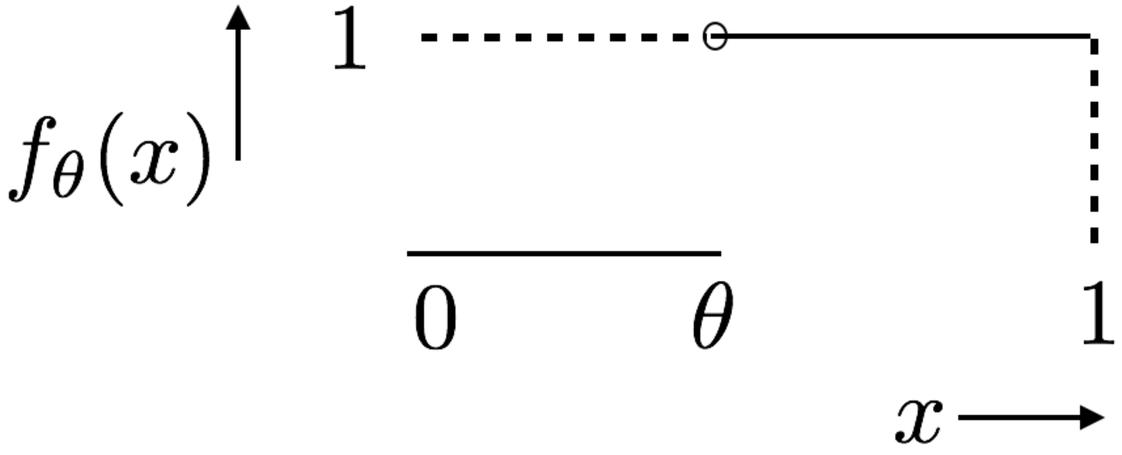

Figure 1: Plot of function $f_\theta(x)$ against x.

Let $T_{min} = \max(\{\frac{1}{4}\} \cup \{X_i | f_\theta(X_i) = 0\})$. We know that the true θ must be larger than T_{min}.

Let $T_{max} = \min(\{\frac{3}{4}\} \cup \{X_i | f_\theta(X_i) = 1\})$. We know that the true θ must be smaller than T_{max}.

The gap between T_{min} and T_{max} represents the uncertainty we will have about the true θ given the training data that we have received.

(a) What is the probability that $T_{max} - \theta > \varepsilon$ as a function of ε.

(b) What is the probability that $\theta - T_{min} > \varepsilon$ as a function of ε.

(c) Suppose that you would like to have an estimate for θ within an accuracy of 2ε with probability at least $1 - \delta$. Please bound or estimate how big of an n do you need?

(d) Let us say that instead of getting random samples $(X_i, f(X_i))$, we were allowed to choose where to sample the function, but you had to choose all the places you were going to sample in advance. Propose a method to estimate θ. How many samples suffice to achieve an estimate that is ε-close (within an interval of size 2ε) as above? (**Hint:** You need not use a randomized strategy.)

(e) Suppose that you could pick where to sample the function adaptively — choosing where to sample the function in response to what the answers were previously. Propose a method to estimate θ. How many samples suffice to achieve an estimate that is ε-close (within an interval of size 2ε) as above?

(f) Compare the scaling of n with ε and δ in the three sampling approaches above: random, deterministic, and adaptive.

4 Eigenvalue and Eigenvector review

A square matrix $A \in \mathbb{R}^{d \times d}$ has a (right) eigenvalue $\lambda \in \mathbb{C}$ and (right) eigenvector $\vec{x} \in \mathbb{C}^d \setminus 0$ if $P\vec{x} = \lambda\vec{x}$. Left eigenvalues and eigenvectors are defined analogously — $\vec{x}^T P = \lambda\vec{x}^T$. Since the definition is scale invariant (if $\vec{x}$ is an eigenvector, then $t\vec{x}$ is an eigenvector for any $t \neq 0$), we adopt the convention that each eigenvector has norm 1.

(a) Compute the right and left eigenvalues and eigenvectors of the following matrices.

i) $A = \begin{bmatrix} 3 & 2 \\ 1 & 3 \end{bmatrix}$

ii) $B = \begin{bmatrix} 5 & 2 \\ 2 & 5 \end{bmatrix}$

iii) A^2

iv) B^2

iv) AB

v) BA

(b) Compute the singular value decompositions of the matrices above. In addition, please compute the SVD of: $C = \begin{bmatrix} 5 & 2 \\ 2 & 5 \\ 3 & 2 \\ 1 & 3 \end{bmatrix}$

(c) Show from the definition of a right eigenvalue that the quantity λ is an eigenvalue with associated eigenvector $\vec{x}$ iff for all $1 \leq i \leq n$, we have

$$(\lambda - A_{ii})x_i = \sum_{j \neq i} A_{ij}x_j.$$

(d) Now for an arbitrary eigenvalue λ of A and its associated eigenvector $\vec{x}$, choose index i such that $|x_i| \geq |x_j|$ for all $j \neq i$. For such an index i, show that

$$|\lambda - A_{ii}| \leq \sum_{j \neq i} |A_{ij}|.$$

You have just proved Gershgorin's circle theorem, which states that all the eigenvalues of a $d \times d$ matrix lie within the union of d disks in the complex plane, where disk i has center A_{ii}, and radius $\sum_{j \neq i} |A_{ij}|$.

5 Fun with least squares

Ordinary least squares for a scalar helps us learn linear predictors for a scalar b given observations $\vec{a}$. This is done by taking our training data points and constructing a tall vector $\vec{b}$ by stacking up our training data's b_i and a corresponding tall matrix A by stacking up our training data's $\vec{a}_i^T$, and then finding the weights $\vec{x}$ that minimize $\|A\vec{x}-\vec{b}\|_2^2$. The resulting weights $\vec{x}$ can be used to predict b's by $\vec{x}^T\vec{a}$.

Let us think about the components of $\vec{a}$ as being a sequence of measurements in time. The first measurement, the second measurement, the third measurement, and so on.

(a) Suppose that you wanted to construct not one linear predictor for b but a sequence of them. One that just used the first measurement. One that used the first two measurements. All the way up to one that uses all the measurements. How would you do this in the most straightforward manner?

(b) Someone suggests that maybe the measurements themselves are partially predictable from the previous measurements. They suggest a two part strategy. First we predict the next measurement based on the measurements so far. And then we look at the difference (sometimes deemed the "innovation") between the actual measurement we made and our prediction for it, and just use that to update our prediction for b.

Give a way to learn (from the training data) these best predictors of the next measurement from the previous measurements as well as learn (from the training data) the weights used to update the prediction for b based on the innovation.

(c) Is this two-part prediction strategy equivalent to the straightforward approach? Why or why not?

HINT: Think about what it might mean to orthonormalize the columns of the training matrix A above.

(d) **(BONUS — but worth doing)** A student complains to you that the case of learning a linear predictor for $\vec{b}$ given $\vec{a}$ seems too complicated. She suggests just setting up the problem directly as learning a matrix X so that $\vec{b} \approx X\vec{a}$. She takes the training data $(\vec{a}_i, \vec{b}_i)$ and stacks them **horizontally** into fat matrices A and B and computes the hypothetical difference matrix $B - XA$.

She would like to minimize an appropriate norm for this and chooses the simple entry-wise 2-norm (also known as the Frobenius norm) squared. So $\min_X \|B - XA\|_F^2$. Recall that the Frobenius norm of a real matrix (need not be square) L is $\|L\|_F^2 = \text{trace}(L^T L)$. Use this and vector calculus to directly solve for the minimizing X.

(e) Comment on why it makes sense that the computation of the best linear prediction of a vector just turns into a set of best linear predictions of each scalar component of the vector being predicted.

6 System Identification by ordinary least squares regression

Making autonomous vehicles involves using machine learning for many purposes. One of which is learning how the car actually behaves based on data about it.

Make sure to submit the code you write in this problem to "HW1 Code" on Gradescope.

(a) Consider a noisy time sequence of scalars $x[t]$. Let $x[t+1] = Ax[t] + Bu[t] + w$ where $A, B \in \mathbb{R}$ and w is the noise. Fit a line $x[t+1] = Ax[t] + Bu[t]$ to the values of x and u as provided by `system_identification_programming_a.mat`.

(b) Consider a noisy time sequence of vectors $x[t]$. Let $x[t+1] = Ax[t] + Bu[t] + w$ where $A, B \in \mathbb{R}^{3\times 3}$ and w is the noise. Fit a line $x[t+1] = Ax[t] + Bu[t]$ to the values of x and u as provided by the `system_identification_programming_b.mat`.

(c) Consider a dynamical system consisting of a cars driving on a straight road. The dynamics (accelerations) for a car i are governed by a linear function of its own position and velocity, as well as the position and velocity of the car $i-1$ preceding it. This is called a linear *car following model*. That is, we may write the dynamical system as follows:

$$\ddot{x}_i = ax_i + b\dot{x}_i + cx_{i-1} + d\dot{x}_{i-1} + w(t)$$

where $w(t) \sim \mathcal{N}(0, \sigma)$. Re-write the dynamical system in matrix form, i.e. in the form $\dot{x} = Ax + Bu + w(t)$.

(d) Given data measurements from a dynamical system, we wish to estimate the parameters of the system. This is called *system identification*. In the car following model example, given vehicle traces consisting of positions, velocities, and accelerations of vehicles and the vehicles preceding them (that is, given samples of $\ddot{x}_i, x_i, \dot{x}_i, x_{i-1}, \dot{x}_{i-1}$, denoted by *row vectors* $\hat{\ddot{x}}_i, \hat{x}_i, \hat{\dot{x}}_i, \hat{x}_{i-1}, \hat{\dot{x}}_{i-1}$, respectively), give the analytical solution to parameters (a, b, c, d) which gives the best linear fit to the data collected from the dynamical system.

(e) Implement your estimator using data file `system_identification_programming_train.mat`, which contains a dictionary with several useful values: `ddot_x_i, x_i, dot_x_i, x_i-1, dot_x_i-1`. There are 40,000 data points. This data is a processed, filtered, and sampled form of data collected from the I-80 highway here in California; it is available from the Next Generation Simulation (NGSIM) dataset, commonly used in traffic simulation. The data is given in units of $m/s^2, m, m/s, m, m/s$, respectively. More precisely, the velocities given are deviations from the speed limit of $29.0576 m/s$ (65mph). Give the resulting parameters below. Describe qualitatively how the dynamics evolve with respect to the state x and input u variables. When are the accelerations positive?

(f) Use your estimated parameters and data file `system_identification_programming_eval.mat` to generate a vehicle trace and submit the evaluation results to Gradescope. You will be given `x_i(0), dot_x_i(0), x_i-1(t), dot_x_i-1(t)` and be asked to submit `x_i(t)` for 150 time steps. The provided data is given at 15Hz and remember that the given velocities are deviations from $29.0576 m/s$.

Instructions: Generate a file named `submission.txt`. Double check that this file is in plaintext. (i.e., `cat submission.txt` does not give you strange characters at the start of your file). Put only one float `x_i(t)` on each line, where the first line is `x_i(0)`. Submit the file to “HW1 Test Set” on Gradescope. You are allowed two submissions per every 24 hours, so start early! If the script catches a formatting error, the submission will **not** count towards your two daily submissions.

7 Your Own Question

Write your own question, and provide a thorough solution.

Writing your own problems is a very important way to really learn material. The famous “Bloom’s Taxonomy” that lists the levels of learning is: Remember, Understand, Apply, Analyze, Evaluate, and Create. Using what you know to create is the top-level. We rarely ask you any HW questions about the lowest level of straight-up remembering, expecting you to be able to do that yourself. (e.g. make yourself flashcards) But we don’t want the same to be true about the highest level.

As a practical matter, having some practice at trying to create problems helps you study for exams much better than simply counting on solving existing practice problems. This is because thinking about how to create an interesting problem forces you to really look at the material from the perspective of those who are going to create the exams.

Besides, this is fun. If you want to make a boring problem, go ahead. That is your prerogative. But it is more fun to really engage with the material, discover something interesting, and then come up with a problem that walks others down a journey that lets them share your discovery. You don’t have to achieve this every week. But unless you try every week, it probably won’t happen ever.