CS/CNS/EE 156a Learning Systems
Caltech - Fall 2016
http://cs156.caltech.edu
(Learning From Data campus version)

Homework # 5

Due Monday, October 31, 2016, at 2:00 PM PDT

*All questions have multiple-choice answers (***[a], [b], [c], ...***). No collaboration except at the edX discussion forum (don't discuss selected or excluded choices in the answers). You can consult books and notes, but not other people's solutions. Your solutions must be based on your own work. Definitions and notation follow the lectures.*

Note about the homework

- Answer each question by selecting from the multiple-choice answers (carries 6 points) **and by giving a brief justification of your selection** (carries 4 points). For experimental problems, include *your own* code and the numerical answer it gave that led to your selection.
- You can select 1 or 2 of the multiple-choice answers for each question, but you will get 6 or 4 points, respectively, for a correct answer.
- The problems range from easy to difficult, and from practical to theoretical. Some problems require running a full experiment to arrive at the answer.
- The answer may not be obvious or numerically close to one of the choices, but one (and only one) choice will be correct if you follow the instructions precisely in each problem. You are encouraged to explore the problem further by experimenting with variations on these instructions, for the learning benefit.
- You are encouraged to take part in the edX discussion forum. Please make sure you don't discuss specific answers, or specific excluded answers, before the homework is due.

• Linear Regression Error

Consider a noisy target $y = \mathbf{w}^{*T}\mathbf{x} + \epsilon$, where $\mathbf{x} \in \mathbb{R}^d$ (with the added coordinate $x_0 = 1$), $y \in \mathbb{R}$, $\mathbf{w}^*$ is an unknown vector, and ϵ is a noise term with zero mean and σ^2 variance. Assume ϵ is independent of $\mathbf{x}$ and of all other ϵ's. If linear regression is carried out using a training data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, and outputs the parameter vector $\mathbf{w}_{\text{lin}}$, it can be shown that the expected in-sample error E_{in} with respect to $\mathcal{D}$ is given by:

$$\mathbb{E}_{\mathcal{D}}[E_{\text{in}}(\mathbf{w}_{\text{lin}})] = \sigma^2 \left(1 - \frac{d+1}{N}\right)$$

1. For $\sigma = 0.1$ and $d = 8$, which among the following choices is the smallest number of examples N that will result in an expected E_{in} greater than 0.008?

[**a**] 10

[**b**] 25

[**c**] 100

[**d**] 500

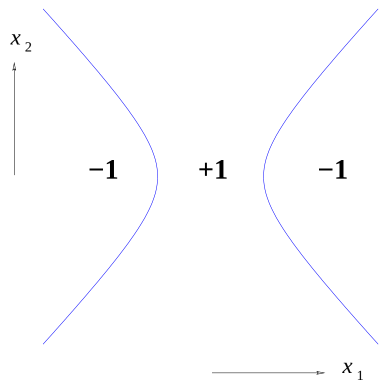
[**e**] 1000

• Nonlinear Transforms

In linear classification, consider the feature transform $\Phi : \mathbb{R}^2 \to \mathbb{R}^2$ (plus the added zeroth coordinate) given by:

$$\Phi(1, x_1, x_2) = (1, x_1^2, x_2^2)$$

2. Which of the following sets of constraints on the weights in the $\mathcal{Z}$ space could correspond to the hyperbolic decision boundary in $\mathcal{X}$ depicted in the figure?



You may assume that $\tilde{w}_0$ can be selected to achieve the desired boundary.

[a] $\tilde{w}_1 = 0, \tilde{w}_2 > 0$

[b] $\tilde{w}_1 > 0, \tilde{w}_2 = 0$

[c] $\tilde{w}_1 > 0, \tilde{w}_2 > 0$

[d] $\tilde{w}_1 < 0, \tilde{w}_2 > 0$

[e] $\tilde{w}_1 > 0, \tilde{w}_2 < 0$

Now, consider the 4th order polynomial transform from the input space $\mathbb{R}^2$:

$$\Phi_4 : \mathbf{x} \to (1, x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3, x_1^4, x_1^3x_2, x_1^2x_2^2, x_1x_2^3, x_2^4)$$

3. What is the smallest value among the following choices that is *not* smaller than the VC dimension of a linear model in this transformed space?

[a] 3

[b] 5

[c] 15

[d] 20

[e] 21

• Gradient Descent

Consider the nonlinear error surface $E(u, v) = (ue^v - 2ve^{-u})^2$. We start at the point $(u, v) = (1, 1)$ and minimize this error using gradient descent in the uv space. Use $\eta = 0.1$ (learning rate, not step size).

4. What is the partial derivative of $E(u, v)$ with respect to u, i.e., $\frac{\partial E}{\partial u}$?

[a] $(ue^v - 2ve^{-u})^2$

[b] $2(ue^v - 2ve^{-u})$

[c] $2(e^v + 2ve^{-u})$

[d] $2(e^v - 2ve^{-u})(ue^v - 2ve^{-u})$

[e] $2(e^v + 2ve^{-u})(ue^v - 2ve^{-u})$

5. How many iterations (among the given choices) does it take for the error $E(u, v)$ to fall below 10^{-14} for the first time? In your programs, make sure to use double precision to get the needed accuracy.

[a] 1

[**b**] 3

[**c**] 5

[**d**] 10

[**e**] 17

6. After running enough iterations such that the error has just dropped below 10^{-14}, what are the closest values (in Euclidean distance) among the following choices to the final (u, v) you got in Problem 5?

[**a**] $(1.000, 1.000)$

[**b**] $(0.713, 0.045)$

[**c**] $(0.016, 0.112)$

[**d**] $(-0.083, 0.029)$

[**e**] $(0.045, 0.024)$

7. Now, we will compare the performance of "coordinate descent." In each iteration, we have two steps along the 2 coordinates. Step 1 is to move only along the u coordinate to reduce the error (assume first-order approximation holds like in gradient descent), and step 2 is to reevaluate and move only along the v coordinate to reduce the error (again, assume first-order approximation holds). Use the same learning rate of $\eta = 0.1$ as we did in gradient descent. What will the error $E(u, v)$ be closest to after 15 full iterations (30 steps)?

[**a**] 10^{-1}

[**b**] 10^{-7}

[**c**] 10^{-14}

[**d**] 10^{-17}

[**e**] 10^{-20}

• Logistic Regression

In this problem you will create your own target function f (probability in this case) and data set $\mathcal{D}$ to see how Logistic Regression works. For simplicity, we will take f to be a 0/1 probability so y is a deterministic function of $\mathbf{x}$.

Take $d = 2$ so you can visualize the problem, and let $\mathcal{X} = [-1, 1] \times [-1, 1]$ with uniform probability of picking each $\mathbf{x} \in \mathcal{X}$. Choose a line in the plane as the boundary between $f(\mathbf{x}) = 1$ (where y has to be $+1$) and $f(\mathbf{x}) = 0$ (where y has to be -1) by taking two random, uniformly distributed points from $\mathcal{X}$ and taking the line passing through them as the boundary between $y = \pm 1$. Pick $N = 100$ training points at random from $\mathcal{X}$, and evaluate the outputs y_n for each of these points $\mathbf{x}_n$.

Run Logistic Regression with Stochastic Gradient Descent to find g, and estimate E_{out} (the **cross entropy** error) by generating a sufficiently large, separate set of points to evaluate the error. Repeat the experiment for 100 runs with different targets and take the average. Initialize the weight vector of Logistic Regression to all zeros in each run. Stop the algorithm when $\|\mathbf{w}^{(t-1)} - \mathbf{w}^{(t)}\| < 0.01$, where $\mathbf{w}^{(t)}$ denotes the weight vector at the end of epoch t. An epoch is a full pass through the N data points (use a random permutation of $1, 2, \cdots, N$ to present the data points to the algorithm within each epoch, and use different permutations for different epochs). Use a learning rate of 0.01.

8. Which of the following is closest to E_{out} for $N = 100$?

[**a**] 0.025
[**b**] 0.050
[**c**] 0.075
[**d**] 0.100
[**e**] 0.125

9. How many epochs does it take on average for Logistic Regression to converge for $N = 100$ using the above initialization and termination rules and the specified learning rate? Pick the value that is closest to your results.

[**a**] 350
[**b**] 550
[**c**] 750
[**d**] 950
[**e**] 1750

• PLA as SGD

10. The Perceptron Learning Algorithm can be implemented as SGD using which of the following error functions $\mathrm{e}_n(\mathbf{w})$? Ignore the points $\mathbf{w}$ at which $\mathrm{e}_n(\mathbf{w})$ is not twice differentiable.

[**a**] $\mathrm{e}_n(\mathbf{w}) = e^{-y_n \mathbf{w}^\intercal \mathbf{x}_n}$
[**b**] $\mathrm{e}_n(\mathbf{w}) = -y_n \mathbf{w}^\intercal \mathbf{x}_n$
[**c**] $\mathrm{e}_n(\mathbf{w}) = (y_n - \mathbf{w}^\intercal \mathbf{x}_n)^2$
[**d**] $\mathrm{e}_n(\mathbf{w}) = \ln(1 + e^{-y_n \mathbf{w}^\intercal \mathbf{x}_n})$
[**e**] $\mathrm{e}_n(\mathbf{w}) = -\min(0, y_n \mathbf{w}^\intercal \mathbf{x}_n)$