CS 5350/6350: Machine Learning Spring 2019

Richard Child u0581030

March 1, 2019

1 Paper Problems [40 points + 6 bonus]

1. [5 points] We have shown that, with probability at least $1-\delta$, a hypothesis $h \in H$ that is consistent with a training set of m examples will have the generalization error $\text{err}_D(h) < \epsilon$ if

$$m > \frac{1}{\epsilon}\big(\log(|H|) + \log\frac{1}{\delta}\big).$$

   (a) [2 points]

      i. [1 point] We prefer the result hypothesis from L_2 since $|H_1| > |H_2|$.

      ii. [1 point] From Occam's Razor we prefer smaller sample complexity m_2, and a smaller $|H|$ results in smaller sample complexity. That is, if ϵ and δ are fixed, and $|H_1| > |H_2|$ then m_1 will need to be greater than m_2 in order to gaurantee PAC learnability.

   (b) [3 points] Using the above inequality

$$m > \frac{1}{0.1}\left(ln(3^{10}) + ln(\frac{1}{0.05})\right)$$
$$\boxed{m > 139.8}$$

2. [7 points]

   (a) [1 points] Simple disjunctions out of n binary variables: yes is PAC learnable. The hypothesis must have error of ϵ, there are $2n$ terms, so each term must have probability of being error of $\frac{\epsilon}{2n}$. Good literals have probability $1-\frac{\epsilon}{2n}$, in terms of m then a bad event has probability $\big(1-\frac{\epsilon}{2n}\big)^m$. Thus $2n(1-\frac{\epsilon}{2n})^m \leq \delta$, and $(1-x) \leq e^{-x}$, so $m \geq (n/\epsilon)ln(n/\delta)$ which is polynomial in n, $1/\epsilon$, and $1/\delta$.

   (b) [1 points] m-of-n rules (Note that m is a fixed constant).

   (c) [1 points] General conjunctions out of n binary variables: no this is NOT PAC learnable. There are 2^n outcomes per example. The error rate for a good prediction would be $(1-\frac{\epsilon}{2^n})^m$, and $m \geq (2^n/\epsilon)ln(n/\delta)$ is NOT polynomial in n.

   (d) [2 points] General Boolean functions of n binary variables.

   (e) [2 points] Can ID3 algorithm *efficiently* PAC learn the above concept classes?

3. [5 points]

$$\epsilon_t = \frac{1}{2} - \frac{1}{2}\Big(\sum_{i=1}^{m} D_t(i) y_i h_t(x_i)\Big)$$

Since $y_i h_t(x_i)$ can only be 1 (correct prediction) or -1 (incorrect prediction), and $D_t(i)$ sums to 1, the edge cases for ϵ_t are as follows:

$$\epsilon_t = \frac{1}{2} - \frac{1}{2}(1) = 0$$
$$\epsilon_t = \frac{1}{2} - \frac{1}{2}(-1) = 1$$

Thus we do not need to consider the correct predictions, just the incorrect ones. So it can be written as $\epsilon_t = \sum_{y_i \neq h_t(x_i)} D_t(i)$.

4. [8 points] Can you figure out an equivalent linear classifier and a decision tree for the following Boolean functions? For linear classifiers, please point out what the weight vector, the bias parameter and the hyperplane are. Note that the hyperplane is determined by an equality. If you cannot find out such a linear classifier, please explain why.

   (a) [1 point] $y = 1$ if $x_1 + x_2 + x_3 \geq 1, w = [1, 1, 1], b = [-1], hplane : -1 + x_1 + x_2 + x_3 = 0$

   (b) [1 point] $y = 1$ if $x_1 - x_2 - x_3 \geq 3, w = [1, -1, -1], b = [-3], hplane : -3 + x_1 - x_2 - x_3 = 0$

   (c) [1 point] $y = 1$ if $-x_1 - x_2 - x_3 \geq 3, w = [-1, -1, -1], b = [-3], hplane : -3 - x_1 - x_2 - x_3 = 0$

   (d) [2 points] There is no linear classifier because it cannot express "if and only if" between two optional statements.

   (e) [2 points] There is no linear classifier because it cannot correctly capture the optional statement combining the two "if and only if" statements.

   (f) [1 point] What do you conclude about the expressiveness of decision trees and linear classifiers? Why?

5. [6 points]

6. [**Bonus**] [6 points]

7. [9 points] Suppose we have the training data shown in Table **??**, from which we want to learn a linear regression model, parameterized by a weight vector $\mathbf{w}$ and a bias parameter b.

   (a) [1 point] $J(\mathbf{w}, b) = \frac{1}{2}\sum_{i=1}^{m}(y_i - [\mathbf{w}, b]^T[x_i, 1])^2$

   (b) [3 points] Calculate the gradient $\frac{\nabla J}{\nabla \mathbf{w}}$ and $\frac{\nabla J}{\nabla b}$

      i. when $\mathbf{w} = [0, 0, 0]^\top$ and $b = 0$, $\nabla J(\mathbf{w}, b) = [-4, 4, -22, -2]$;

      ii. when $\mathbf{w} = [-1, 1, -1]^\top$ and $b = -1$, $\nabla J(\mathbf{w}, b) = [-20, 14, -56, -12]$;

      iii. when $\mathbf{w} = [1/2, -1/2, 1/2]^\top$ and $b = 1$, $\nabla J(\mathbf{w}, b) = [17/2, -5, -5, 7/2]$.

   (c) [2 points] What are the optimal $\mathbf{w}$ and $\mathbf{b}$ that minimize the cost function?

   (d) [3 points] Now, we want to use stochastic gradient descent to minimize $J(\mathbf{w}, b)$. We initialize $\mathbf{w} = \mathbf{0}$ and $b = 0$. We set the learning rate $r = 0.1$ and sequentially go through the 5 training examples. Please list the stochastic gradient in each step and the updated $\mathbf{w}$ and b.

x_1	x_2	x_3	y
1	-1	2	1
1	1	3	4
-1	1	0	-1
1	2	-4	-2
3	-1	-1	0

Table 1: Linear regression training data.

2 Practice [60 points + 10 bonus]

1. [2 Points] Update your machine learning library. Please check in your implementation of decision trees in HW1 to your GitHub repository. Remember last time you created a folder "Decision Tree". You can commit your code into that folder. Please also supplement README.md with concise descriptions about how to use your code to learn decision trees (how to call the command, set the parameters, etc). Please create two folders "Ensemble Learning" and "Linear Regression" in the same level as the folder "Decision Tree".

2. [36 points]

   (a) [8 points]

   (b) [8 points]

   (c) [6 points]

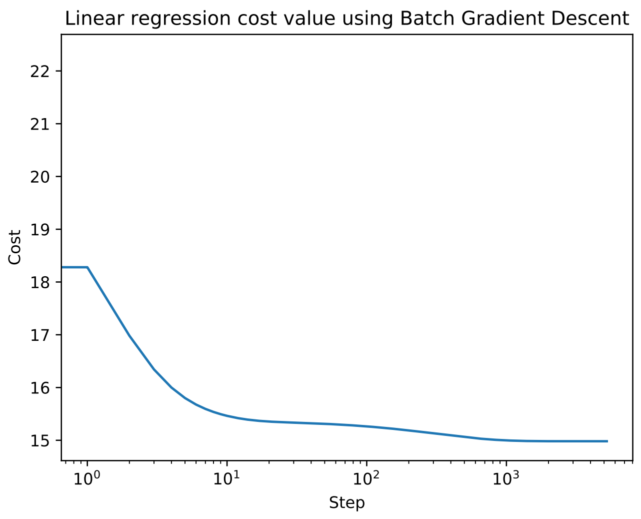
   (d) [8 points]

   (e) [6 points]

3. [**Bonus**][10 points]

4. [22 points]

   (a) [8 points] The learned weight vector is $[0.9003, 0.7860, 0.8508, 1.2987, 0.1299, 1.5719, 0.9984, -0.0152]$ and the learning rate is 0.75. The costs for each step of the gradient descent are recorded in the file *concrete_4a_results.txt*. Using the final weight vector on the test data yields a cost of 23.36. The following figure is of *Cost* vs. *Steps* generated from the Batch Gradient Descent on the training data.



(b) [8 points]

(c) [6 points]