COMS 4771 FA18 HW1

Due: Fri Oct 05, 2018 at 11:59pm

You are allowed to write up solutions in groups of (at max) three students. These group members don't necessarily have to be the same from previous homeworks. Only one submission per group is required by the due date on Gradescope. Name and UNI of all group members must be clearly specified on the homework. No late homeworks are allowed. To receive credit, a typesetted copy of the homework pdf must be uploaded to Gradescope by the due date. You must show your work to receive full credit. Discussing possible solutions for homework questions is encouraged on piazza and with peers outside your group, but every group must write their own individual solutions. You must cite all external references you used (including the names of individuals you discussed the solutions with) to complete the homework.

1 **[Statistical Estimators]** Here we will study some statistical estimators.

(i) Given $a, b \in \mathbb{R}$ s.t. $a < b$, consider the density $p(x \mid \theta = (a, b)) \propto \begin{cases} 1 & \text{if } a \leq x \leq b \\ 0 & \text{otherwise} \end{cases}$.
Suppose that n samples $x_1, \ldots, x_n$ are drawn i.i.d. from $p(x|\theta)$. What is the Maximum Likelihood Estimate (MLE) of θ given the samples?

(ii) Show that for the MLE θ_{ML} of a parameter $\theta \in \mathbb{R}^d$ and any known differentiable function $g : \mathbb{R}^d \to \mathbb{R}^k$, the MLE of $g(\theta)$ is $g(\theta_{\text{ML}})$.

(iii) For a 1-dimensional Gaussian distribution, give two examples for each of the following types of estimators for the mean parameter.
- consistent and unbiased.
- consistent, but not unbiased.
- not consistent, but unbiased.
- neither consistent, nor unbiased.

2 **[Maximum Likelihood Estimation (MLE) versus Maximum a Posteriori (MAP) Estimation]** Here we investigate the difference between MLE vs. MAP estimation using a specific example.
Your friend gives you a coin with bias b (that is, tossing the coin turns '1' with probability b, and turns '0' with probability $1 - b$). You make n independent tosses and get the observation sequence $x_1, \ldots, x_n \in \{0, 1\}$.

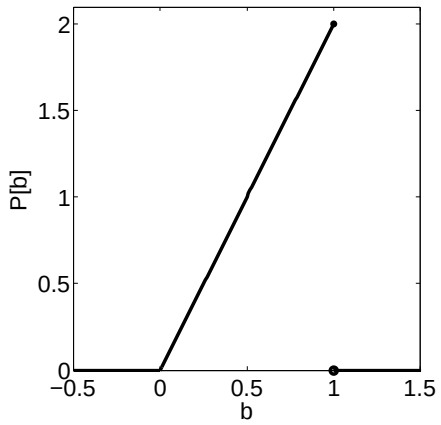
(i) You want to estimate the coin's bias. What is the Maximum Likelihood Estimate (MLE) $\hat{b}$ given the observations $x_1, \ldots, x_n$?

(ii) Is your estimate from part (i) an unbiased estimator of b? How about consistent? Justify your answer.

(iii) Derive a simple expression for the variance of this coin?

(iv) What is the MLE for the coin's variance?

(v) Your friend reveals to you that the coin was minted from a faulty press that biased it towards 1. Suppose the model for the faulty bias is given by the following distribution: Having this extra knowledge, what is the best estimate for the coin's bias b given the



observation sequence? That is, compute: $\arg\max_b P[b \mid x_1, \ldots, x_n]$.

Note: this estimate of the coin's bias encorporates prior knowledge, and is call a MAP estimate.

(vi) When does MAP estimate equals MLE?

3 **[Designing the optimal predictor for continuous output spaces]** We studied in class that the "Bayes Classifier" $\big(f := \arg\max_y P[Y|X]\big)$ is optimal in the sense that it minimizes generalization error over the underlying distribution, that is, it maximizes $\mathbb{E}_{x,y}[\mathbf{1}[g(x) = y]]$. But what can we say when the output space $\mathcal{Y}$ is continuous?

Consider predictors of the kind $g : \mathcal{X} \to \mathbb{R}$ that predict a real-valued output for a given input $x \in \mathcal{X}$. One intuitive way to define the quality of of such a predictor g is as

$$Q(g) := \mathbb{E}_{x,y}[(g(x) - y)^2].$$

Observe that one would want a predictor g with the lowest $Q(g)$.

(i) Show that if one defines the predictor as $f(x) := \mathbb{E}[Y|X = x]$, then $Q(f) \leq Q(g)$ for any g, thereby showing that f is the optimal predictor with respect to Q for continuous output spaces.

(ii) If one instead defines quality as $Q(g) := \mathbb{E}_{x,y}\big[|g(x) - y|\big]$, which f is the optimal predictor? Justify your reasoning.

4 **[Finding (local) minima of generic functions]** Finding extreme values of functions in a closed form is often not possible. Here we will develop a generic algorithm to find the extremal values of a function. Consider a smooth function $f : \mathbb{R} \to \mathbb{R}$.

(i) Recall that Taylor's Remainder Theorem states:
For any $a, b \in \mathbb{R}$, exists $z \in [a, b]$, such that $f(b) = f(a) + f'(a)(b-a) + \frac{1}{2}f''(z)(b-a)^2$.

Assuming that there exists $L \geq 0$ such that for all $a, b \in \mathbb{R}$, $|f'(a) - f'(b)| \leq L|a - b|$, prove the following statement:

For any $x \in \mathbb{R}$, there exists some $\eta > 0$, such that if $\bar{x} := x - \eta f'(x)$, then $f(\bar{x}) \leq f(x)$, with equality if and only if $f'(x) = 0$.

(Hint: first show that the assumption implies that f has bounded second derivative, i.e., $f''(z) \leq L$ (for all z); then apply the remainder theorem and analyze the difference $f(x) - f(\bar{x})$).

(ii) Part (i) gives us a generic recepie to find a new value $\bar{x}$ from an old value x such that $f(\bar{x}) \leq f(x)$. Using this result, develop an iterative algorithm to find a local minimum starting from an initial value x_0.

(iii) Use your algorithm to find the minimum of the function $f(x) := (x - 4)^2 + 2e^x$. You should code your algorithm in a scientific programming language like Matlab to find the solution.

5 **[Training error vs. Test error]** In this problem we will study why we expect training error to be smaller than test error.

Suppose $(x_1, y_1), \ldots, (x_n, y_n), (x, y)$ are i.i.d. samples taking values in $\mathbb{R}^d \times \mathbb{R}$. Let $\mathcal{R}$ denote the squared error of a (linear) classifier, defined as

$$\mathcal{R}(w) := \mathbb{E}\big[(w \cdot x - y)^2\big]$$

for any $w \in \mathbb{R}^d$, and let $\hat{\mathcal{R}}$ denote the training error based on $(x_1, y_1), \ldots, (x_n, y_n)$, so $\hat{\mathcal{R}}(w) := \frac{1}{n}\sum_{i=1}^n (w \cdot x_i - y_i)^2$ for any $w \in \mathbb{R}^d$.

Let $\hat{w}$ denote the squared training error minimizing decision boundary based on samples $(x_1, y_1), \ldots, (x_n, y_n)$, so $\hat{\mathcal{R}}(\hat{w}) \leq \hat{\mathcal{R}}(w)$ for all $w \in \mathbb{R}^d$. (You may assume that $\hat{w}$ is unique.) Prove that

$$\mathbb{E}\Big[\hat{\mathcal{R}}(\hat{w})\Big] \leq \mathbb{E}\Big[\mathcal{R}(\hat{w})\Big],$$

where the expectation on both sides is taken with respect to $(x_1, y_1), \ldots, (x_n, y_n)$.

Hint: Let $(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_n, \tilde{y}_n)$ be another i.i.d. random sample, independent of $(x_1, y_1), \ldots, (x_n, y_n)$, but having the same distribution as (x, y). Then

$$\mathcal{R}(w) = \mathbb{E}\Big[\underbrace{\frac{1}{n}\sum_{i=1}^n (w \cdot \tilde{x}_i - \tilde{y}_i)^2}_{=:\tilde{\mathcal{R}}(w)}\Big], \quad w \in \mathbb{R}^d.$$

Now let $\tilde{w}$ denote the squared training error minimizing decision boundary based on samples $(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_n, \tilde{y}_n)$ (again, assume uniqueness). How do $\mathbb{E}[\tilde{\mathcal{R}}(\hat{w})]$, $\mathbb{E}[\tilde{\mathcal{R}}(\tilde{w})]$, and $\mathbb{E}[\hat{\mathcal{R}}(\hat{w})]$ compare?

6 **[Email spam classification case study]** Download the datafile hw1data.tar.gz. This datafile contains email data of around 5,000 emails divided in two folders 'ham' and 'spam' (there are about 3,500 emails in the 'ham' folder, and 1,500 emails in the 'spam' folder). Each email is a separate text file in these folders. These emails have been slightly preprocessed to remove meta-data information.

(i) (Embedding text data in Euclidean space) The first challenge you face is how to systematically embed text data in a Euclidean space. It turns out that one successful way of transforming text data into vectors is via "Bag-of-words" model. Basically, given a dictionary of all possible words in some order, each text document can be represented as a word count vector of how often each word from the dictionary occurs in that document.

Example: suppose our dictionary D with vocabulary size 10 ($|D| = 10$). The words (ordered in say alphabetical order) are:

1: also

2: football

3: games

4: john

5: likes

6: Mary

7: movies

8: to

9: too

10: watch

Then any text document created using this vocabulary can be embedded in $\mathbb{R}^|D|$ by counting how often each word appears in the text document.

Say, an example text document t is:

```
John likes to watch football.  Mary likes movies.
```

Then the corresponding word count vector in $|D| = 10$ dimensions is:

```
[ 0 1 0 1 2 1 1 1 0 1]
```

(because the word "also" occurs 0 times, "football" occurs 1 time, etc. in the document.)

While such an embedding is extremely useful, a severe drawback of such an embedding is that it treats similar meaning words (e.g. watch, watches, watched, watching, etc.) independently as separate coordinates. To overcome this issue one should preprocess the entire corpus to remove the common trailing forms (such as "ing", "ed", "es", etc.) and get only the root word. This is called word-stemming.

Your first task is to embed the given email data in a Euclidean space by: first performing word stemming, and then applying the bag-of-words model.

Some useful references:

- Bag-of-words: http://en.wikipedia.org/wiki/Bag-of-words_model
- Word stemming: http://en.wikipedia.org/wiki/Stemming

(ii) Once you have a nice Euclidean representation of the email data. Your next task is to develop a spam classifier to classify new emails as `spam` or `not-spam`. You should compare performance of naive-bayes, nearest neighbor (with L_1, L_2 and L_∞ metric) and decision tree classifiers.

(you may use builtin functions for performaing basic linear algebra and probability calculations but you should write the classifiers from scratch.)

You must submit your code to Courseworks to receive full credit.

(iii) Which classifier (discussed in part (ii)) is better for the email spam classification dataset? You must justify your answer with appropriate performance graphs demonstrating the superiority of one classifier over the other. Example things to consider: you should evaluate how the classifier behaves on a holdout 'test' sample for various splits of the data; how does the training sample size affects the classification performance.