COMS 4771 HW1 (Fall 2022)

Due: Fri Oct 07, 2022 at 11:59pm

This homework is to be done **alone**. 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 approaches for solutions for homework questions is encouraged on the course discussion board and with your peers, but you must write your own individual solutions and **not** share your written work/code. You must cite all resources (including online material, books, articles, help taken from/given to specific individuals, etc.) you used to complete your work.

1 Maximum Likelihood Estimation

https://math.stackexchange.com/questions/946368/find-the-maximum-likelihood-estimator-of-theta

(a) Consider the density $p(x \mid \theta) \coloneqq \begin{cases} e^{\theta - x} & \text{if } x \geq \theta \\ 0 & \text{otherwise} \end{cases}$ for some $\theta > 0$. Suppose that n samples $x_1, x_2, \ldots, x_n > 0$ are drawn i.i.d. from $p(x \mid \theta)$. What is the MLE of θ given the samples?

Hint: In class we computed the MLE by finding the log-likelihood, taking the gradient (with respect to θ), and setting it equal to 0. However, this only works well for differentiable concave log-likelihood functions. You should never forget that the end goal is simply to maximize the likelihood (or log-likelihood) function.

(b) Show that for the MLE θ_{MLE} of a parameter $\theta \in \mathbb{R}^d$ and any known bijective function g, the MLE of $g(\theta)$ is $g(\theta_{\text{MLE}})$. It turns out that this holds for any function g, i.e. g need not be bijective (you do not need to show this but you should try to understand why this is true). Use this fact to find the MLE of e^θ in the setting of part (a).

(c) Sometimes we have prior knowledge concerning the value of the parameter θ. This is often encoded as a prior distribution characterized by $p(\theta)$. In such cases one usually computes the MAP (maximum a posteriori) estimate of θ, defined as $\theta_{\text{MAP}} = \arg\max_\theta p(\theta \mid x_1, x_2, \ldots, x_n)$, instead of the MLE.

In the setting of part (a) suppose that we know from prior information that θ is likely small. Specifically, we model this with the prior $p(\theta) = 2e^{-\theta^2}\pi^{-1/2}$. Compute the MAP estimate of θ.

Hint: First show that $\theta_{\text{MAP}} = \arg\max_\theta p(x_1, x_2, \ldots, x_n \mid \theta)p(\theta)$.

2 Analyzing iterative optimization

Minimizing an objective function is of central importance in machine learning. In this problem, we will analyze the an iterative approach for finding $\beta \in \mathbb{R}^d$ that (approximately) minimizes $\|A\beta - b\|_2^2$ for a given matrix $A \in \mathbb{R}^{n \times d}$ and vector $b \in \mathbb{R}^n$.

Consider the following iterative approximation algorithm:

- Initially, $\beta^{(0)} = (0, \ldots, 0) \in \mathbb{R}^d$ is the zero vector in $\mathbb{R}^d$.
- For $k = 1, 2, \ldots, N$:
  - Compute $\beta^{(k)} := \beta^{(k-1)} + \eta A^\mathsf{T}(b - A\beta^{(k-1)})$.

In above, $\eta > 0$ is a fixed positive number usually referred to as the step size, and N is the total number of iterations. Define $M := A^\mathsf{T}A$ and $v := A^\mathsf{T}b$.

(i) Show that the matrix M is symmetric positive semi-definite.

Throughout, assume that the eigenvalues of M, denoted by $\lambda_1, \ldots, \lambda_d$, satisfy $\lambda_i < 1/\eta$ for all $i = 1, \ldots, d$.

(ii) Prove (e.g., using mathematical induction) that, for any positive integer N,

$$\beta^{(N)} = \eta \sum_{k=0}^{N-1} (I - \eta M)^k v.$$

(Here, for a square matrix B, we have $B^0 = I$, $B^1 = B$, $B^2 = BB$, $B^3 = BBB$, and so on.)

(iii) What are the eigenvalues of $\eta \sum_{k=0}^{N-1} (I - \eta M)^k$? Give your answer in terms of $\lambda_1, \ldots, \lambda_d$, η, and N.

(iv) Let $\hat{\beta}$ be any non-zero vector in the range of M satisfying $M\hat{\beta} = v$. Prove that

$$\|\beta^{(N)} - \hat{\beta}\|_2^2 \leq e^{-2\eta\lambda_{\min} N} \|\hat{\beta}\|_2^2,$$

where $\lambda_{\min}$ is the smallest non-zero eigenvalue of M.

Hint: You may use the fact that $1 + x \leq e^x$ for any $x \in \mathbb{R}$.

This implies that as the number of iterations N increases, the difference between our estimate $\beta^{(N)}$ and $\hat{\beta}$ decreases exponentially!

3 3 and 5-Nearest Neighbor Analysis

Show that the asymptotic error rate of 3-NN classifier is at most 1.6 times Bayes optimal classifier. What is the best 5-NN asymptotic error rate you can get (wrt Bayes error rate)?

4 Email spam classification case study

Download the datafile `email_data.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.