COMS 4771 HW1 (Fall 2021)

Due: Oct 08, 2021 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 solutions for homework questions is encouraged on the course discussion board and with your peers, but you must write their own individual solutions. You should cite all resources (including online material, books, articles, help taken from specific individuals, etc.) you used to complete your work.

1 Analyzing Bayes Classifier

Consider a binary classification problem where the output variable $Y \in \{0, 1\}$ is fully determined by variables A, B and C (each in $\mathbb{R}$). In particular

$$Y = \begin{cases} 1 & \text{if } A + B + C < 7 \\ 0 & \text{otherwise} \end{cases}.$$

1. Let each A, B and C be i.i.d. exponential random variable (with mean parameter $\lambda = 1$).
   (a) Suppose the variables A and B are known but C is unknown, compute $P[Y = 1|A, B]$, the optimal Bayes classifier, and the corresponding Bayes error.
   (b) If only the variable A is known but the variables B and C are unknown, compute $P[Y = 1|A]$, the optimal Bayes classifier, and the corresponding Bayes error.
   (c) If none of the variables A, B, C are known, what is the Bayes classifier and the corresponding Bayes error?
2. Assume that variables A, B and C are independent. For known A and B, show that there exists a distribution on C for which the Bayes classification error rate can be made as close to $1/2$ as desired.

2 3-Nearest Neighbor Analysis

Show that the asymptotic error rate of 3-NN classifier is at most 1.6 times Bayes optimal classifier.

3 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 recipe 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 of your choice to find the solution. (You don't need to submit your code for this part)

4 Designing socially aware classifiers

Traditional Machine Learning research focuses on simply improving the accuracy. However, the model with the highest accuracy may be discriminatory and thus may have undesirable social impact that unintentionally hurts minority groups[1]. To overcome such undesirable impacts, researchers have put lots of effort in the field called Computational Fairness in recent years.

Two central problems of Computational Fairness are: (1) what is an appropriate definition of fairness that works under different settings of interest? (2) How can we achieve the proposed definitions without sacrificing on prediction accuracy?

In this problem, we will focus on some of the ways we can address the first problem. There are two categories of fairness definitions: individual fairness[2] and group fairness[3]. Most works in the literature focus on the group fairness. Here we will study some of the most popular group fairness definitions and explore them empirically on a real-world dataset.

Generally, group fairness concerns with ensuring that group-level statistics are same across all groups. A group is usually formed with respect to a feature called the **sensitive attribute**. Most common sensitive features include: gender, race, age, religion, income-level, etc. Thus, group

[1] see e.g. **Machine Bias** by Angwin et al. for bias in recidivism predication, and **Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification** by Buolamwini and Gebru for bias in face recognition

[2] see e.g. **Fairness Through Awareness** by Dwork et al.

[3] see e.g. **Equality of Opportunity in Supervised Learning** by Hardt et al.

fairness ensures that statistics across the sensitive attribute (such as across, say, different age groups) remain the same.

For simplicity, we only consider the setting of binary classification with a single sensitive attribute. Unless stated otherwise, we also consider the sensitive attribute to be binary. (Note that the binary assumption is only for convenience and results can be extended to non-binary cases as well.)

Notations:

Denote $X \in \mathbb{R}^d,\ A \in \{0,1\}$ and $Y \in \{0,1\}$ to be three random variables: non-sensitive features of an instance, the instance's sensitive feature and the target label of the instance respectively, such that $(X, A, Y) \sim \mathcal{D}$. Denote a classifier $f : \mathbb{R}^d \to \{0,1\}$ and denote $\hat{Y} := f(X)$.

For simplicity, we also use the following abbreviations:

$$\mathbb{P} := \mathbb{P}_{(X,A,Y)\sim D} \qquad \text{and} \qquad \mathbb{P}_a := \mathbb{P}_{(X,a,Y)\sim D}$$

We will explore the following are three fairness definitions.

- Demographic Parity (DP)

$$\mathbb{P}_0[\hat{Y} = \hat{y}] = \mathbb{P}_1[\hat{Y} = \hat{y}] \qquad \forall \hat{y} \in \{0,1\}$$

(equal positive rate across the sensitive attribute)

- Equalized Odds (EO)

$$\mathbb{P}_0[\hat{Y} = \hat{y} \mid Y = y] = \mathbb{P}_1[\hat{Y} = \hat{y} \mid Y = y] \qquad \forall \hat{y},\ y \in \{0,1\}$$

(equal true positive- and true negative-rates across the sensitive attribute)

- Predictive Parity (PP)

$$\mathbb{P}_0[Y = y \mid \hat{Y} = \hat{y}] = \mathbb{P}_1[Y = y \mid \hat{Y} = \hat{y}] \qquad \forall \hat{y},\ y \in \{0,1\}$$

(equal positive predictive- and negative predictive-value across the sensitive attribute)

Part 0: The basics.

(i) Why is it not enough to just remove the sensitive attribute A from the dataset to achieve fairness as per the definitions above? Explain with a concrete example.

Part 1: Sometimes, people write the same fairness definition in different ways.

(ii) Show that the following two definitions for *Demographic Parity* is equivalent under our setting:

$$\mathbb{P}_0[\hat{Y} = 1] = \mathbb{P}_1[\hat{Y} = 1] \iff \mathbb{P}[\hat{Y} = 1] = \mathbb{P}_a[\hat{Y} = 1] \qquad \forall a \in \{0,1\}$$

(iii) Generalize the result of the above equivalence and state an analogous equivalence relationship of two equality when $A \in \mathbb{N}$, and $\hat{Y} \in \mathbb{R}$.

Part 2: In this part, we will explore the COMPAS dataset. The task is to predict two year recidivism. In COMPAS dataset, the target label Y is `two_year_recid` and the sensitive feature A is `race`.

(iv) Develop the following classifiers: (1) MLE based classifier, (2) nearest neighbor classifier, and (3) naïve-bayes classifier, for the given dataset.

For MLE classifier, you can model the class conditional densities by a Multivariate Gaussian distribution. For nearest neighbor classifier, you should consider different values of k and the distance metric (e.g. L_1, L_2, L_∞). For the naïve-bayes classifier, you can model the conditional density for each feature value as count probabilities.

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

You must submit your code on Gradescope to receive full credit.

(v) Which classifier (discussed in previous part) is better for this prediction task? You must justify your answer with appropriate performance graphs demonstrating the superiority of one classifier over the other. Example things to consider: how does the training sample size affects the classification performance.

(vi) To what degree the fairness definitions are satisfied for each of the classifiers you developed? Show your results with appropriate performance graphs.

For each fairness measure, which classifier is the most fair? How would you summarize the difference of these algorithms?

(vii) Choose any one of the three fairness definitions. Describe a real-world scenario where this definition is most reasonable and applicable. What are the potential disadvantage(s) of this fairness definition?

(You are free to reference online and published materials to understand the strengths and weaknesses of each of the fairness definitions. Make sure cite all your resources.)

(viii) [Optional problem, will not be graded] Can an algorithm simultaneously achieve high accuracy and be fair and unbiased on this dataset? Why or why not, and under what fairness definition(s)? Justify your reasoning.

5 A comparative study of classification performance of handwritten digits

Download the datafile `digits.mat`. This datafile contains 10,000 images (each of size 28x28 pixels = 784 dimensions) of handwritten digits along with the associated labels. Each handwritten digit belongs to one of the 10 possible categories $\{0, 1, \ldots, 9\}$. There are two variables in this datafile: (i) Variable X is a 10,000x784 data matrix, where each row is a sample image of a handwritten digit.

(ii) Variable Y is the 10,000x1 label vector where the i^{th} entry indicates the label of the i^{th} sample image in X.

Special note for those who are not using Matlab: Python users can use `scipy` to read in the mat file, R users can use `R.matlab` package to read in the mat file, Julia users can use `JuliaIO/MAT.jl`. Octave users should be able to load the file directly.

To visualize this data (in Matlab): say you want to see the actual handwritten character image of the 77th datasample. You may run the following code (after the data has been loaded):

```
figure;
imagesc(1-reshape(X(77,:),[28 28])');
colormap gray;
```

To see the associated label value:

```
Y(77)
```

(i) Create a probabilistic classifier (as discussed in class) to solve the handwritten digit classification problem. The class conditional densities of your probabilistic classifier should be modeled by a Multivariate Gaussian distribution. It may help to recall that the MLE for the parameters of a Multivariate Gaussian are:

$$\vec{\mu}_{\text{ML}} := \frac{1}{n}\sum_{i=1}^{n}\vec{x}_i$$

$$\Sigma_{\text{ML}} := \frac{1}{n}\sum_{i=1}^{n}(\vec{x}_i - \vec{\mu}_{\text{ML}})(\vec{x}_i - \vec{\mu}_{\text{ML}})^{\mathsf{T}}$$

You must submit your code on Gradescope to receive full credit.

(ii) Create a k-Nearest Neighbor classifier (with Euclidean distance as the metric) to solve the handwritten digit classification problem.

You must submit your code on Gradescope to receive full credit.

(iii) Which classifier (the one developed in Part (i) or the one developed in Part (ii)) is better? 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.

(iv) As discussed in class, there are several metrics one can use in a Nearest Neighbor classification. Do a similar analysis to justify which of the three metrics: L_1, L_2 or L_∞ is better for handwritten digit classification problem.

Note: All plots, analysis and results for this question should be included in the pdf document. No credit will be awarded if the plots and analysis is not in the pdf document.