August 3, 2021
Changho Suh (chsuh@kaist.ac.kr)

Lecture 3: A fair classifier using kernel density estimation

Outline

In this lecture, we will study another fair classifier [2] that addresses the training instability issue occurring for the MI-based fair classifier [1]. Specifically this lecture consists of five parts. First we will revisit the MI-based optimization. We will then explore a way to directly compute the fairness measure DDP, instead of employing mutual information. Next, we will introduce a trick that allows us to well approximate DDP. The trick is based on a well-known technique prevalent in statistics and information theory, named *Kernel Density Estimation (KDE)*. In the fourth part, we will develop a KDE-based optimization for a fair classifier. Lastly we will study how to solve the optimization.

Revisit: MI-based optimization

Recall the MI-based optimization:

$$\min_{w} \frac{1-\lambda}{m} \sum_{i=1}^{m} \ell_{\mathsf{CE}}(y^{(i)}, \hat{y}^{(i)}) + \lambda \cdot I(Z; \hat{Y}). \tag{1}$$

Here the key rationale behind the training instability problem is that mutual information $I(Z;\hat{Y})$ is represented as "max" optimization, thus incurring the "min max" structure that often suffers from training instability.

Go back to the fairness measure DDP

In an effort to address the problem, let us start from the beginning, observing the fairness measure DDP again:

$$\mathsf{DDP} := \sum_{z \in \mathcal{Z}} |\mathbb{P}(\tilde{Y}=1|Z=z) - \mathbb{P}(\tilde{Y}=1)|. \tag{2}$$

There are two important probabilities for each absolute function that constitutes the summation. Let us first focus on the second probability of a simpler form and try to compute the probability in a *direct* manner (without relying upon other measures like mutual information):

$$\mathbb{P}(\tilde{Y}=1) = \mathbb{P}(\hat{Y} \geq \tau) = \int_{\tau}^{\infty} f_{\hat{Y}}(t)dt \tag{3}$$

where the 1st equality is due to $\tilde{Y} := \mathbf{1}\{\hat{Y} \geq \tau\}$. Here $f_{\hat{Y}}(t)$ denotes the probability density function (pdf) of $\hat{Y}$. An issue occurs in computing this probability. The problem is that the pdf $f_{\hat{Y}}(t)$ is *unknown*. Instead, we are given only samples $\{\hat{y}^{(1)}, \ldots, \hat{y}^{(m)}\}$. A natural question arises. Is there a way to infer the pdf only from such samples?

Kernel density estimation

It turns out kernel density estimation (KDE) comes to rescue. To see this clearly, let us first figure out what the KDE is. Given samples $\{\hat{y}^{(1)}, \ldots, \hat{y}^{(m)}\}$, the KDE is defined as:

$$\widehat{f}_{\hat{Y}}(t) := \frac{1}{mh} \sum_{i=1}^{m} f_{\mathsf{ker}}\left(\frac{t-\hat{y}^{(i)}}{h}\right) \tag{4}$$

where $f_{\mathsf{ker}}(\cdot)$ indicates a kernel function (e.g., Gaussian kernel) and h denotes a parameter called *bandwidth* subject to our design choice. The parameter is also named as a smoothing parameter, as its magnitude affects the smoothness of the estimated pdf curve $\widehat{f}_{\hat{Y}}(t)$.

Here one crucial thing to worry about is *accuracy* of the KDE. How accurate is the KDE in particular w.r.t. the number m of samples? There have been lots of works on analyzing the scaling behavior of the difference between the ground-truth pdf and its KDE estimate. One recent analysis is due to Jiang [3] who showed that

$$\left|\widehat{f}(t) - f(t)\right|_{\infty} \lesssim \frac{1}{m^{\frac{1}{d}}} \tag{5}$$

where d is the dimension of an interested random variable. This suggests that m should grow *exponentially* in the dimension d for ensuring fast enough convergence to the ground-truth pdf. The estimate may not be accurate under high-dimensional settings. Actually this is one of the main reasons as to why the KDE is not heavily employed for many high-dimensional applications. However, a good news comes in the classifier setting that we focus on herein. The good news is that $d = 1$ in our setting; hence, the approximation via KDE is moderately good even for a not-super-large value of m.

Approximation via KDE

The observation based on (5) naturally motivates us to approximate (3) via KDE. Specifically we employ a Gaussian kernel function:

$$f_{\mathsf{ker}}(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2} \qquad \forall t \in \mathbb{R}. \tag{6}$$

Plugging the KDE-approximated pdf $\widehat{f}_{\hat{Y}}(t)$ into (3) in place of $f_{\hat{Y}}(t)$, we obtain:

$$\begin{aligned}
\widehat{\mathbb{P}}(\tilde{Y}=1) &= \int_{\tau}^{\infty} \widehat{f}_{\hat{Y}}(t)dt \\
&\overset{(a)}{=} \int_{\tau}^{\infty} \frac{1}{mh} \sum_{i=1}^{m} f_{\mathsf{ker}}\left(\frac{t-\hat{y}^{(i)}}{h}\right) dt \\
&\overset{(b)}{=} \frac{1}{m} \sum_{i=1}^{m} \int_{\frac{\tau-\hat{y}^{(i)}}{h}}^{\infty} f_{\mathsf{ker}}\left(y\right) dy \\
&\overset{(c)}{=} \frac{1}{m} \sum_{i=1}^{m} Q\left(\frac{\tau-\hat{y}^{(i)}}{h}\right)
\end{aligned} \tag{7}$$

where (a) comes from the use of KDE (4); (b) is due to the change of variable $y := \frac{t-\hat{y}^{(i)}}{h}$; and (c) is because we employ the Q-function: $Q(z) := \int_z^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2} dt$.

Remember in the definition (2) of DDP that we have another probability $\mathbb{P}(\tilde{Y}=1|Z=z)$ inside the summation. Applying the same trick, one can obtain:

$$\widehat{\mathbb{P}}(\tilde{Y}=1|Z=z) = \frac{1}{m_z} \sum_{i \in I_z} Q\left(\frac{\tau-\hat{y}^{(i)}}{h}\right) \tag{8}$$

where $I_z := \{i : z^{(i)} = z\}$ and $m_z := |I_z|$. Here the only distinction is that we concern only samples subject to $z^{(i)} = z$. The derivation is not that difficult. Please check it if you are not convinced.

Approximated DDP

We are now ready to approximate DDP (2). Applying (7) and (8) into (2) instead of the ground-truth probabilities, we get:

$$\begin{aligned}
\mathsf{DDP} &:= \sum_{z\in\mathcal{Z}} |\mathbb{P}(\tilde{Y}=1|Z=z) - \mathbb{P}(\tilde{Y}=1)| \\
&\approx \sum_{z\in\mathcal{Z}} |\widehat{\mathbb{P}}(\tilde{Y}=1|Z=z) - \widehat{\mathbb{P}}(\tilde{Y}=1)| \\
&= \sum_{z\in\mathcal{Z}} \left| \frac{1}{m_z} \sum_{i\in I_z} Q\left(\frac{\tau-\hat{y}^{(i)}}{h}\right) - \frac{1}{m}\sum_{i=1}^{m} Q\left(\frac{\tau-\hat{y}^{(i)}}{h}\right) \right| \\
&\approx \sum_{z\in\mathcal{Z}} \left| \frac{1}{m_z} \sum_{i\in I_z} e^{-\frac{(\tau-\hat{y}^{(i)})^2}{2h^2}} - \frac{1}{m}\sum_{i=1}^{m} e^{-\frac{(\tau-\hat{y}^{(i)})^2}{2h^2}} \right|
\end{aligned} \tag{9}$$

where the approximation in the last step follows from the well-known approximation of the Q-function: $Q(x) \approx e^{-\frac{x^2}{2}}$. Notice that DDP is now an explicit function of the samples $\hat{y}^{(i)}$'s and therefore it can well be expressed in terms of the classifier parameter w.

KDE-based optimization [2]

Employing the approximated DDP (9) in the fairness-regularized optimization, we obtain:

$$\min_{w} \frac{1-\lambda}{m}\sum_{i=1}^{m} \ell_{\mathsf{CE}}(y^{(i)},\hat{y}^{(i)}) + \frac{\lambda}{m}\cdot \sum_{z\in\mathcal{Z}} \left| \frac{m}{m_z} \sum_{i\in I_z} e^{-\frac{(\tau-\hat{y}^{(i)})^2}{2h^2}} - \sum_{i=1}^{m} e^{-\frac{(\tau-\hat{y}^{(i)})^2}{2h^2}} \right| \tag{10}$$

where $0 \le \lambda \le 1$ indicates a regularization factor that we interpret as the fairness tuning knob. Since it is well expressed in terms of w, one can readily resort to a famous gradient descent algorithm to solve the optimization. However, there are some issues in solving the optimization. Two issues. One is how to deal with the absolute function that leads to non-differentiability. The second is how to choose the smoothing parameter h called bandwidth.

To resolve the first issue, one can employ Huber loss instead of the absolute function:

$$H_\delta(x) = \begin{cases} \frac{1}{2}x^2, & \text{if } |x| \le \delta; \\ \delta\left(|x| - \frac{1}{2}\delta\right), & \text{otherwise.} \end{cases} \tag{11}$$

Notice that when x is around 0, the function is indeed differentiable while respecting the original linear behavior when x is apart from 0. This then enables us to readily obtain gradient.

Regarding the second issue, it turns out there is a sweet spot on h in light of the mean square error of KDE estimate. This advises us to find h^* that minimizes the mean square error. The computation of h^* is involved and hence we omit the detailed derivation in this tutorial; see the supplementary in [2] if you are interested in.

Extension to another fairness measure DEO

Similarly one can apply the same KDE trick to another fairness measure DEO, thus approximating it as:

$$\begin{aligned}
\mathsf{DEO} &:= \sum_{y\in\mathcal{Y}}\sum_{z\in\mathcal{Z}} |\mathbb{P}(\tilde{Y}=1|Y=y,Z=z) - \mathbb{P}(\tilde{Y}=1|Y=y)| \\
&\approx \sum_{y\in\mathcal{Y}}\sum_{z\in\mathcal{Z}} |\widehat{\mathbb{P}}(\tilde{Y}=1|Y=y,Z=z) - \widehat{\mathbb{P}}(\tilde{Y}=1|Y=y)| \\
&\approx \sum_{y\in\mathcal{Y}}\sum_{z\in\mathcal{Z}} \left| \frac{1}{m_{yz}} \sum_{i\in I_{yz}} e^{-\frac{(\tau-\hat{y}^{(i)})^2}{2h^2}} - \frac{1}{m_y}\sum_{i\in I_y} e^{-\frac{(\tau-\hat{y}^{(i)})^2}{2h^2}} \right|
\end{aligned} \tag{12}$$

where $I_{yz} := \{i : y^{(i)} = y, z^{(i)} = z\}$ and $m_{yz} := |I_{yz}|$. Here the only distinctions are on the summations inside the absolute function. We aggregate associated probabilities for only the samples subject to $\{y^{(i)} = y, z^{(i)} = z\}$ or $\{y^{(i)} = y\}$.

Experiments

We provide experimental results for two classifiers (MI-based and KDE-based fair classifiers) to demonstrate that the KDE-based fair classifier offers training stability while yielding a better accuracy-vs-DDP tradeoff. We consider the same benchmark real dataset: COMPAS [4]. Fig. 1

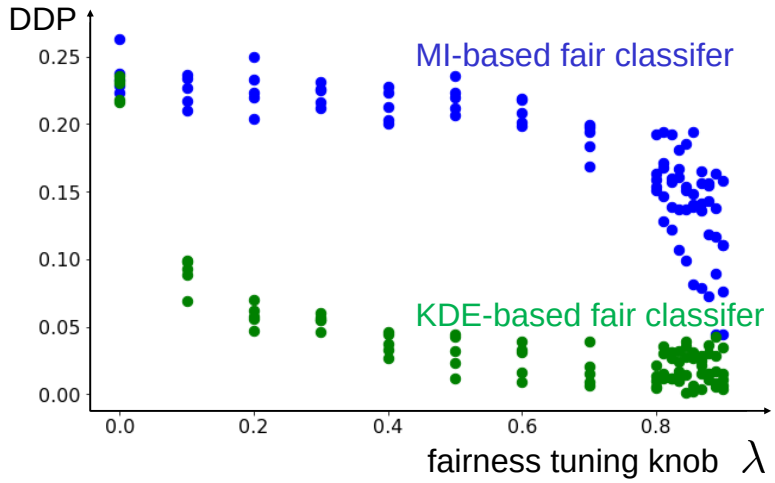

Figure 1: DDP as a function of the fairness tuning knob λ for two approaches: (blue) MI-based fair classifier; (green) KDE-based fair classifier. Each blue-or-green dot corresponds to a single result w.r.t. one particular seed for training. While the blue dots are quite spread near $\lambda \approx 1$, the green dots are more concentrated, thus offering training stability. The case of $\lambda \approx 1$ is also a practically relevant regime especially when putting a strong emphasis on fairness.

plots the DDP performance as a function of the fairness tuning knob λ for the two classifiers. Notice that the green dots are more concentrated, thus offering training stability.

Accuracy vs DDP tradeoff

We also provide accuracy-vs-DDP tradeoff performances for the two classifiers in Fig. 2. Unlike the setting in Fig. 1, here each dot corresponds to the *average* result over five trials of training with distinct seeds. We see that the KDE-based fair classifier offers a greater tradeoff as well. Actually it is not 100% obvious why the KDE-based fair classifier outperforms the MI-based one although it directly computes DDP. This is because the KDE approach allows us to *only approximate* DDP (not exactly). Unfortunately, there is no theoretical proof for that. So you can just consider this as sort of an experimental support.

Summary of Lectures 1/2/3

So far we have explored fairness issues that arise in the context of classifiers. Specifically we investigated two major fairness measures: DDP and DEO. We then made an interesting connection between the fairness measures and mutual information (MI). Building upon the connection, we next investigated an MI-based fair classifier which offers a good tradeoff yet suffering from training instability. Finally we studied another fair classifier based on KDE, which addresses

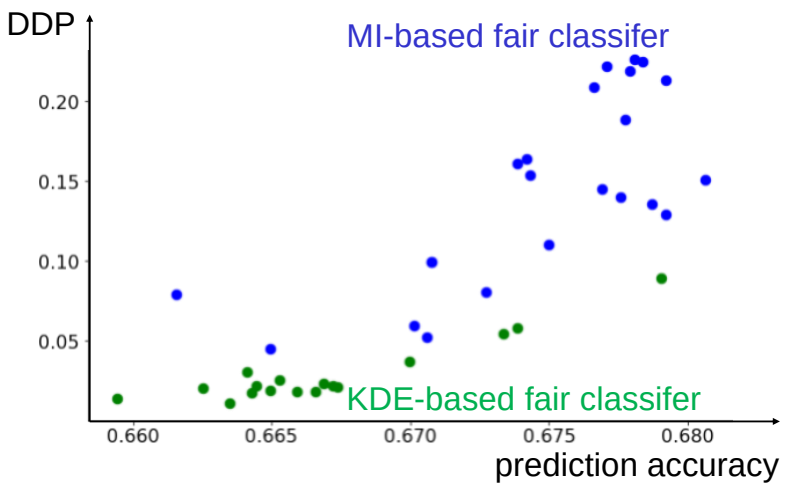

Figure 2: Accuracy vs DDP tradeoff performances for two approaches: (blue) MI-based fair classifier; (green) KDE-based fair classifier. Here each dot represents the *average* over five trials of training with different seeds. The KDE-based fair classifier yields a greater tradeoff.

the training instability issue.

Look ahead

This is the end of today's morning session. Tomorrow morning, we will move onto the second context that I promised to cover in Lecture 1. The context is about *fair generative models*. Specifically we will study a prominent generative model, named hinge GAN, which forms the basis of a fair generative model to be explored in depth (Lecture 4). We will then make an interesting connection with a very well-known divergence in statistics and information theory, *total variation distance (TVD)*. Building upon the connection, we will develop a TVD-based fair generative model (Lecture 5). In the last lecture, we will conclude the tutorial with a couple of discussions regarding both fairness and robustness aspects, as well as regarding other contexts beyond fair classifiers and fair generative models.

References

[1] J. Cho, G. Hwang and C. Suh. A fair classifier using mutual information. *IEEE International Syposium on Inofrmation Theory (ISIT)*, 2020.

[2] J. Cho, G. Hwang and C. Suh. A fair classifier using kernel density estimation. *In Advances in Neural Information Processing Systems 33 (NeurIPS)*, 2020.

[3] H. Jiang. Uniform convergence rates for kernel density estimation. *International Conference on Machine Learning (ICML)*, 2017.

[4] J. Angwin, J. Larson, S. Mattu, and L. Kirchner. Machine bias: There's software used across the country to 272 predict future criminals. And it's biased against blacks. https://www.propublica.org/article/machine-bias-risk-assessments-incriminal-sentencing, 2015.