Data Imputation using Generative Adversarial Nets

Yunzhen Yao
ShanghaiTech University
Shanghai, China
yaoyzh@shanghaitech.edu.cn

1. Introduction

Missing data is a common problem - data is missing because it was never collected, records were lost or for many other reasons. In the medical domain, the respiratory rate of a patient may not have been measured (perhaps because it was deemed unnecessary/unimportant) or accidentally not recorded. It may also be the case that certain pieces of information are difficult or even dangerous to acquire (such as information gathered from a biopsy). An imputation algorithm can be used to estimate missing values based on data that was observed/measured, such as the systolic blood pressure and

The main focus of this work is to design a missing data imputation method estimating missing values based on data that was observed/measured. Existing imputation methods can be categorized as either discriminative or generative. Discriminative methods include MICE, MissForest, and matrix completion; generative methods include algorithms based on Expectation Maximization and algorithms based on deep learning, for example, denoising autoencoders (DAE) and generative adversarial nets (GAN). However, current generative methods for imputation have various drawbacks. Some need to make assumptions about the underlying distribution and fails to generalize well when datasets contain mixed categorical and continuous variables. Methods using DAE generalize have been shown to work well in practice but require complete data during training, while in many circumstances, missing values are part of the inherent structure of the problem so obtaining a complete dataset is impossible.

In this work we want to design a novel imputation method, which generalizes the well-known GAN and is able to operate successfully even when complete data is unavailable. In this method, the generator's goal is to accurately impute missing data, and the discriminator's goal is to distinguish between observed and imputed components. The discriminator is trained to minimize the classification loss (classifying which components were observed and which have been imputed), and the generator is trained to maximize the discriminator's misclassification rate. Thus, these two networks are trained using an adversarial process. To achieve this goal, we make use of the standard GAN architecture and basing on which we do some adaption. To ensure that the result of this adversarial process is the desired target, the architecture provides the discriminator with additional information in the form of "hints". This hinting ensures that the generator generates samples according to the true underlying data distribution.

2. Problem statement

Consider a d-dimensional space $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_d$. Suppose that $\mathbf{X} = (X_1, \ldots, X_d)$ is a random variable (either continuous or binary) taking values in $\mathcal{X}$ whose distribution we will denote $P(\mathbf{X})$. Suppose that $\mathbf{M} = (M_1, \ldots, M_d)$ is a random variable taking values in $\{0,1\}^d$. We will call $\mathbf{X}$ the data vector, and $\mathbf{M}$ the mask vector.

For each $i \in \{1, \ldots, d\}$, we define a new space $\tilde{\mathcal{X}}_i = \mathcal{X}_i \cup \{*\}$ where $*$ is simply a point not in any $\mathcal{R}_i$, representing an unobserved value. Let $\tilde{\mathcal{X}} = \tilde{\mathcal{X}}_1 \times \ldots \times \tilde{\mathcal{X}}_d$. We define a new random variable $\tilde{\mathbf{X}} = \left(\tilde{X}_1, \ldots, \tilde{X}_d\right) \in \tilde{\mathcal{X}}$ in the following way:

$$\tilde{X}_i = \begin{cases} X_i, & \text{if } M_i = 1 \\ *, & \text{otherwise} \end{cases} \tag{1}$$

so that $\mathbf{M}$ indicates which components of $\mathcal{X}$ are observed. Note that we can recover $\mathbf{M}$ from $\tilde{\mathbf{X}}$.

In the imputation setting, n i.i.d. copies of $\tilde{\mathbf{X}}$ are realized, denoted $\tilde{\mathbf{x}}^1, \ldots, \tilde{\mathbf{X}}^n$ and we define the dataset $\mathcal{D} = \left\{\left(\tilde{\mathbf{x}}^i, \mathbf{m}^i\right)\right\}_{i=1}^n$, where m^i is simply the recovered realization of $\mathbf{M}$ corresponding to $\tilde{\mathbf{x}}^i$.

Our goal is to impute the unobserved values in each $\tilde{\mathbf{x}}^i$. Formally, we want to generate samples according to $P(\mathbf{X}|\tilde{\mathbf{X}} = \tilde{\mathbf{x}}^i)$, the conditional distribution of $\mathcal{X}$ given $\tilde{\mathbf{X}} = \tilde{\mathbf{x}}^i$, for each i, to fill in the missing data points in $\mathcal{D}$. By attempting to model the distribution of the data rather than just the expectation, we are able to make multiple draws and therefore make multiple imputations allowing us to capture the uncertainty of the imputed values.

3. Technical Approach

In this section we describe our approach for simulating $P(\mathbf{X}|\tilde{\mathbf{X}} = \tilde{\mathbf{x}}^i)$ which is motivated by GANs. We highlight key similarities and differences to a standard (conditional) GAN throughout.

3.1. Generator

The generator, G, takes (realizations of) $\tilde{\mathbf{X}}$, $\mathbf{M}$ and a noise variable, $\mathbf{Z}$, as input and outputs $\bar{X}$, a vector of imputations. Let $G : \tilde{\mathcal{X}} \times \{0,1\}^d \times [0,1]^d \to \mathcal{X}$ be a function, and $\mathbf{Z} = (Z_1, \ldots, Z_d)$ be d-dimensional noise (independent of all other variables).

Then we define the random variables $\bar{\mathbf{X}}, \hat{\mathbf{X}} \in \mathcal{X}$ by

$$\bar{\mathbf{X}} = G(\tilde{\mathbf{X}}, \mathbf{M}, (1 - \mathbf{M}) \odot \mathbf{Z}) \tag{2}$$

$$\hat{\mathbf{X}} = \mathbf{M} \odot \tilde{\mathbf{X}} + (1 - \mathbf{M}) \odot \bar{\mathbf{X}} \tag{3}$$

where $\odot$

denotes element-wise multiplication. $\bar{\mathbf{X}}$ corresponds to the vector of imputed values (note that G outputs a value for every component, even if its value was observed) and $\hat{\mathbf{X}}$ corresponds to the completed data vector, that is, the vector obtained by taking the partial observation $\tilde{\mathbf{X}}$ and replacing each $*$ with the corresponding value of $\bar{\mathbf{X}}$.

3.2. Discriminator

As in the GAN framework, we introduce a discriminator, D, that will be used as an adversary to train G. However, unlike in a standard GAN where the output of the generator is either completely real or completely fake, in this setting the output is comprised of some components that are real and some that are fake. Rather than identifying that an entire vector is real or fake, the discriminator attempts to distinguish which components are real (observed) or fake (imputed) - this amounts to predicting the mask vector, $\mathbf{m}$.

Formally, the discriminator is a function $D : \mathcal{X} \to [0,1]^d$ with the i-th component of $D(\hat{\mathbf{x}})$ corresponding to the probability that the i-th component of $\hat{x}$ was observed.

3.3. Hint

A hint mechanism is a random variable, $\mathbf{H}$, taking values in a space $\mathcal{H}$, both of which we define. We allow $\mathbf{H}$ to depend on $\mathbf{M}$ and for each (imputed) sample $(\hat{\mathbf{x}}, \mathbf{m})$, we draw $\mathbf{h}$ according to the distribution $\mathbf{H}|\mathbf{M} = \mathbf{m}$. We pass $\mathbf{h}$ as an additional input to the discriminator and so it becomes a function $D : \mathcal{X} \times \mathcal{H} \to [0,1]^d$, where now the i-th component of $D(\hat{\mathbf{x}}, \mathbf{h})$ corresponds to the probability that the i-th component of $\hat{\mathbf{x}}$ was observed conditional on $\hat{\mathbf{X}} = \hat{\mathbf{x}}$ and $\mathbf{H} = \mathbf{h}$.

4. Intermediate/Preliminary Results

None.