Filling Survey Missing Data With Generative Adversarial Imputation Networks

Cesar Y. Villarreal Guzman

09 December 2020

Abstract

In this paper we propose an alternative method for dealing with missing values on survey data sets. The method leverages two known techniques, categorical encoding and generative adversarial imputation networks. We test this approach on the "Kaggle Data Science Survey" and the "Stack Overflow Annual Developer Survey", we experiment with different proportions of missing values and sample sizes and find the technique to yield high quality data imputations.

Keywords: Generative Adversarial Networks; Imputation Algorithms; Surveys

1. Introduction

Survey response rates have seen a decline in recent years and more often than not we see incomplete survey data sets. Often this is because survey designers give the option to skip a question, or simply a respondent decides to not finish the survey. In the literature this is known as data missing completely at random (MCAR) because in most cases there is no dependency on any of the variables. This pervasive problem has also been the cause for multiple solutions to emerge. An imputation algorithm, for example, can be used to estimate missing values based on data that was observed/measured. A substantial amount of research has been dedicated to developing imputation algorithms for medical data but it is also commonly used in image concealment, data compression, and counterfactual estimation.

Often imputation algorithms work very well when the data is continuous, and or contains a mixture of continuous and categorical responses, however it is common to only observe categorical and text responses in survey data sets. Text data is an almost impossible problem to solve because we can't just simply create an algorithm that will write an opinion on behalf of another person. There are both ethical and technical problems associated. Categorical responses on the other hand are simpler to use because having a finite amount of categories allows us to encoded the data. The most popular encoding technique is known in the statistics literature as dummy variable encoding or in the computer science and machine learning literature as one-hot encoding. This popular technique also comes with its limitations since a substantial amount of information is lost by turning variables into vectors of 0 and 1s.

Moreover this technique requires us to increase the dimensions of our data set which results in a loss of computational efficiency.

Hence, we address the problem of data imputation when the data set consists of only categorical variables, and in particular when the data comes from a survey. In this paper we propose an alternative method for data imputation in survey data which comprises of combining two known methods, categorical encoding and a state of the art imputation algorithm. Specifically, we encode categorical variables with a technique based on the weight of evidence (WOE) method and then use the imputation algorithm known as generative adversarial imputation networks (GAIN) to fill missing values.

The paper is divided in the following manner, first in section 2 we elaborate on generative adversarial imputation networks and how they are applied in this context by discussing the proposed encoding technique based on the weight of evidence method. In section 3 we discuss the experiments we conducted on the "*Kaggle Data Science Survey*" and the "*Stack Overflow Annual Developer Survey*" to test the effectiveness of this method. In this section we comment on the surveys, network architectures and hyperparameters and our empirical results. Finally, in the last section we comment on our results and the implications, how this method could be applied in practice, limitations and areas of future work.

2. Survey Generative Adversarial Imputation Networks

2.1 Generative Adversarial Imputation Networks

First to understand generative adversarial imputation networks (GAIN) we comment on the GAN framework. Generative adversarial nets (GANs) define a framework for estimating generative models via an adversarial process in which two models are trained: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G (Goodfellow et al. 2014). Commonly both the generator and discriminator are two separate neural networks.

Generative adversarial imputation networks (GAIN) is an imputation algorithm that generalized the idea of traditional GANs. The generator's goal is to accurately impute the missing data and the discriminator's goal is to distinguish between observed and imputed components. The discriminator is trained to minimize the classification loss (when classifying which components were observed and which have been imputed), and the generator is trained to maximize the discriminator's misclassification rate (Yoon, Jordon, and Schaar 2018). As with regular GANs both networks are trained in an adversarial process. In the following sections we give a brief explanation of how the GAIN framework is applied, we advice the reader to look at Yoon, Jordon, and Schaar (2018) for the theoretical details.

2.1.1 Data Imputation Problem

To understand what GAINs do we first need to explain what the problem is, to do so let us introduce some notation. Consider the random variable $\mathbf{X} = (X_1, \ldots, X_d)$, called the data vector, which takes values in a d-dimensional space V^d, and a random variable

$\mathbf{M} = (M_1, \ldots, M_d)$, called the mask vector, taking values in $\{0,1\}^d$. For each $i \in \{1, \ldots, d\}$ we define a new space $\tilde{V} = V \cup \{NaN\}$, where the variable NaN represents a point not in any V_i which is an unobserved value. Let $\tilde{V}^d = \tilde{V}_1 \times \cdots \times \tilde{V}_d$ and define a new random variable $\tilde{\mathbf{X}} \in \tilde{V}^d$ in the following way

$$\tilde{\mathbf{X}} = \begin{cases} X_i & \text{if } M_i = 1 \\ NaN & \text{otherwise} \end{cases} \tag{1}$$

i.e. the random variable $\mathbf{M}$ indicates which entries of $\mathbf{X}$ are observed in $\tilde{\mathbf{X}}$. Suppose we have n i.i.d copies $\tilde{\mathbf{X}}$ denoted $\tilde{\mathbf{x}}^1, \ldots, \tilde{\mathbf{x}}^n$ then we define the data set $\mathcal{D} = \{(\tilde{\mathbf{x}}^i, \mathbf{m}^i)\}_{i=1}^n$ where each $\mathbf{m_i}$ indicates with a 0 the values missing in $\tilde{\mathbf{x}}^i$. The goal of data imputation and hence the problem we address is that of modeling $P(\mathbf{X}|\tilde{\mathbf{X}} = \tilde{\mathbf{x}}^i)$.

2.1.2 GAIN Methodology

Given data $\mathcal{D} = \{(\tilde{\mathbf{x}}^i, \mathbf{m}^i)\}_{i=1}^n$ as described above consider the function $G : \tilde{V}^d \times \{0,1\}^d \times [0,1]^d \to V^d$ called the generator which takes as input $\tilde{\mathbf{x}}^i$, $\mathbf{m}^i$ and a noise vector $\mathbf{z} \in [0,1]^d$ of the same dimension as $\tilde{\mathbf{x}}^i$. This noise vector is sampled independently of $\tilde{\mathbf{x}}^i$ and $\mathbf{m}^i$. We denote the vector of imputed values and the completed data vector respectively as

$$\bar{\mathbf{x}}^i = G(\tilde{\mathbf{x}}^i, \mathbf{m}^i, (\mathbf{1} - \mathbf{m}^i) \odot \mathbf{z}) \tag{2}$$

$$\hat{\mathbf{x}}^i = \mathbf{m}^i \odot \tilde{\mathbf{x}}^i + (\mathbf{1} - \mathbf{m}^i) \odot \bar{\mathbf{x}}^i \tag{3}$$

where $\mathbf{1}$ denotes a vector of 1s and $\odot$ represents element wise multiplication. A function $D : V^d \times \mathcal{H} \to [0,1]^d$, called the discriminator, takes as input completed vectors $\hat{\mathbf{x}}^i$ and has the objective of attempting to distinguish which components are real (observed) or fake (imputed) - this amounts to predicting the mask vector, $\mathbf{m}^i$. In particular the j-th entry of $D(\hat{\mathbf{x}}^i, \mathbf{h})$ denotes the probability that the j-th entry of $\hat{\mathbf{x}}^i$ was observed. The vector $\mathbf{h}$ is what the authors of GAIN refer to as the hint mechanism which is a matrix that resembles the true mask $\mathbf{m}^i$ but has a number of differences that "helps" the discriminator D predict the true mask $\mathbf{m}$. Figure 1 was taken directly from the original GAIN paper and helps understand what was mentioned in this section more intuitively.

The objective is as follows: we train D to maximize the probability of correctly predicting $\mathbf{M}$ and G to minimize the probability of D predicting $\mathbf{M}$. Notice the training is adversarial which resembles the original GAN framework. We define the quantity $V(D, G)$ to be

$$V(G, D) = \mathbb{E}_{\hat{\mathbf{X}}, \mathbf{M}, \mathbf{H}} \left[\mathbf{M}^T \log D(\hat{\mathbf{X}}, \mathbf{H}) + (\mathbf{1} - \mathbf{M})^T \log(\mathbf{1} - D(\hat{\mathbf{X}}, \mathbf{H})) \right] \tag{4}$$

where log is element wise and dependence on G is trough $\hat{\mathbf{X}}$. The objective can then be described in notation as

$$\min_G \max_D V(D, G) \tag{5}$$

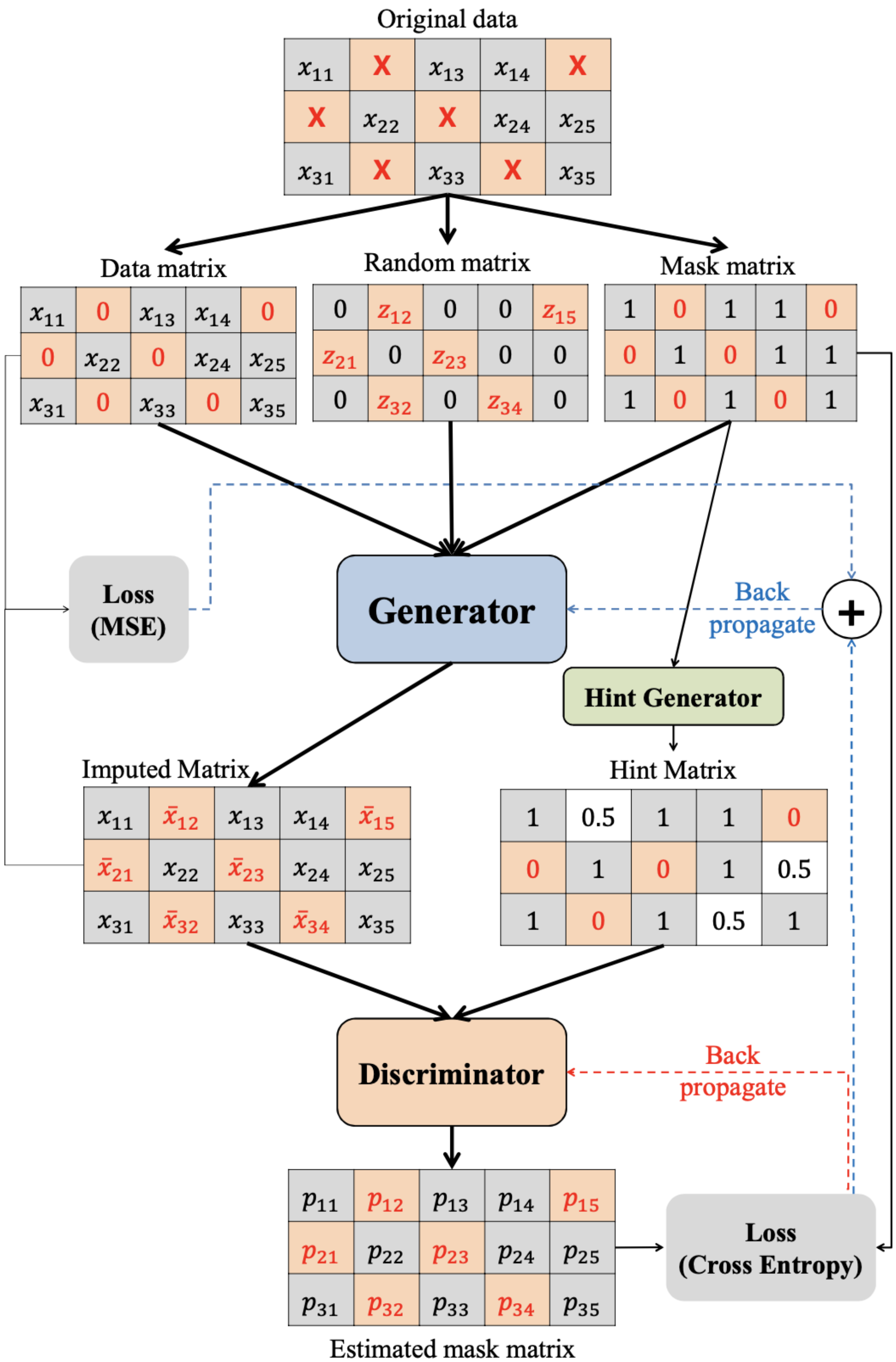

Figure 1: GAIN Architecture

which is the known min-max problem introduced by the GAN framework. In practice the optimization problem we solve is as follows: let $\hat{\mathbf{M}} = D(\hat{\mathbf{X}}, \mathbf{H})$ then the optimization problem can be re-written to

$$\mathcal{L}(\mathbf{M}, \hat{\mathbf{M}}) = \sum_{j=1}^{d} M_j \log(\hat{M}_j) + (1 - M_j)\log(1 - \hat{M}_j) \tag{6}$$

$$\min_G \max_D \mathbb{E}\left[\mathcal{L}(\mathbf{M}, \hat{\mathbf{M}})\right] \tag{7}$$

2.2 Variable Encoding

In practice there are several techniques to encode categorical variables into a continuous or numerical variable. The weight of evidence is one such technique which evolved from the logistic regression framework and has been used in the credit scoring world for decades (Bhalla 2015). Since it evolved from credit scoring world, it is generally described as a measure of the separation of good and bad customers. This technique is great when used to calculate the information value (IV) of a variable, which quantifies the predictive capacity. However in the context of generative networks we do not have a target variable as with a typical machine learning setting. Hence we derived a very similar formula to that of the weight of evidence which we will call it simply as the *weight* of category c.

We define the weight of a category c as follows

$$W(c) = \log\left(\frac{\text{\# of non-}c\text{'s in the data}}{\text{\# of }c\text{'s in the data}}\right) \tag{8}$$

Here log denotes the natural logarithm. In the event that two categories result in the same count then vary the count of one of these variables by 1. That is, if possible then either add or subtract 1 from the count of one of the variables to avoid a collision.

2.3 GAINs on Survey Data

3. Experiment

3.1 Data

3.2 Pre-processing and Network Architecture

3.3 Results

4. Discussion

References

Bhalla, Deepanshu. 2015. "Weight of Evidence (Woe) and Information Value (Iv) Explained." https://www.listendata.com/2015/03/weight-of-evidence-woe-and-information.html.

Goodfellow, Ian J., Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. "Generative Adversarial Networks." http://arxiv.org/abs/1406.2661.

R Core Team. 2020. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686. https://doi.org/10.21105/joss.01686.

Xie, Yihui. 2020. *Knitr: A General-Purpose Package for Dynamic Report Generation in R*. https://yihui.org/knitr/.

Yoon, Jinsung, James Jordon, and Mihaela van der Schaar. 2018. "GAIN: Missing Data Imputation Using Generative Adversarial Nets." http://arxiv.org/abs/1806.02920.