Methods for handling missing and categorical data for classification with neural networks*

Jason Poulos† Rafael Valle‡

Abstract

Researchers analyzing survey data typically choose decision trees or random forests for prediction tasks, largely because missing data and categorical variables are not easy to handle with neural networks. This paper investigates techniques for handling missing data and encoding categorical data such that it is appropriate to neural network classifiers. We experiment on the Adult dataset $N = 48,842$, comparing six different imputation strategies, a simple and complex neural network classifier, and a 3x3 parameter grid. We select three cross–validated models to predict on the test data and find the simple neural network trained on data with missing values imputed using random forests yields the lowest generalization error.

1. Introduction

Missing data is a common problem in survey data in various domains. Several techniques for data imputation (i.e., replace missing values with plausible ones) and direct estimation (i.e., all missing data is analyzed using a maximum likelihood approach) have been proposed [3].

Random Forests and other ensembles of decision trees are the method of choice for survey data, largely because missing data and categorical variables are not easy to handle with neural networks. We investigate techniques for handling missing data and encoding categorical data such that it is appropriate to neural network classifiers. We compare six different imputation strategies: case substitution; mean or median imputation; one–hot; hot deck and cold deck; prediction model; and factor analysis. These strategies are defined in Section 2.1.

After briefly reviewing related works in Section 2, we experiment using neural networks on benchmark data and compare our results with the state-of-the-art in Section 3. Finally, we draw conclusions in Section 4.

2. Related work

2.1. Techniques for handling missing data

We categorize proposed imputation methods into six groups listed below [1]:

Case substitution One observation with missing data is replaced with another non-sampled observation.

Summary statistic Replace the missing data with the mean, median, or mode of the feature vector. Using a numerical approach directly is not appropriate for nonordinal categorical data.

One-hot Create a binary variable to indicate whether or not a specific feature is missing.

Hot deck and cold deck Compute the K-Nearest Neighbors of the observation with missing data and assign the mode of the K-neighbors to the missing data. algorithm.

Prediction Model Train a prediction model (e.g., random forests) to predict the missing value.

Factor analysis Perform factor analysis (e.g., principal component analysis (PCA)) on the design matrix, project the design matrix onto the first two eigenvectors and replace the missing values by the values that might be given by the projected design matrix.

†poulos@berkeley.edu. SID: 24993379.

‡rafaelvalle@berkeley.com. SID: 24090989.

*The video presentation can be viewed at https://youtu.be/2bSPE6gzbN8. The code used for this project is available at https://github.com/jvpoulos/cs289-project.

2.2. Neural networks for classification with categorical and continuous features

Common techniques for handling categorical data in neural networks include encoding the categorical values into numeric values or using binary encoding. These techniques, however, have some drawbacks including unnecessarily increasing model complexity or feature dimensionality and not preserving the similarity information embedded between categorical values [5].

More elaborate techniques include information theoretic measures [10], training separate output units for each of the allowed combination of values of the categorical independent variables [2], and using distance hierarchies [5].

In the case of categorical variables, which by definition have no direct representation or computation scheme of the distance between its values, decision trees can be useful because they do not require distance metrics. However, their training process is slow given a large enough dataset and they might not be suitable for problems where the decision boundary between classes described by a second-order polynomial,* for example [4].

3. Experiments

3.1. Benchmark data set

We experiment on the Adult dataset from the UCI Machine Learning Repository [7]. The dataset has $N = 48,842$ instances, 2/3 for training and 1/3 reserved as a final test set (i.e., train = $32,561$ and test = $16,281$). The dataset contains 14 features: 6 continuous and 8 categorical. The prediction task is to determine whether a person makes over $50,000 a year. 24% of individuals in the training data make more than this amount.

Table 1 shows the test error rates obtained by the data set donor [6]. All error rates were obtained after removing samples with missing values. The state–of–the–art is a Naive Bayes classifier that achieves a 14.05% error rate.

3.2. Patterns of missing values in Adult dataset

The Adult dataset has 3,620 (7.4%) samples containing missing values. Missing values occur in three of the categorical features: *Work class*, *Occupation*, and *Native country*. It is unlikely that these values are missing completely at random (MCAR); it is more likely, and much less desirable that the values are not missing at random (MNAR). Since these data originate from a survey, the missing values may be due to respondents unwilling or unable to provide an answer.

Algorithm	Error (%)
1 C4.5	15.54
2 C4.5-auto	14.46
3 C4.5 rules	14.94
4 Voted ID3 (0.6)	15.64
5 Voted ID3 (0.8)	16.47
6 T2	16.84
7 1R	19.54
8 NBTree	14.10
9 CN2	16.00
10 HOODG	14.82
11 FSS Naive Bayes	14.05
12 IDTM (Decision table)	14.46
13 Naive-Bayes	16.12
14 Nearest-neighbor (1)	21.42

Table 1. Test set error rates on Adult dataset for various algorithms, obtained after removal of samples with missing values and using the original train/test split. Source: [7].

Uncovering patterns of missing values in the dataset will help select strategies for imputing missing values. The histogram (left) in Figure 1 shows *Work class* and *Occupation* each have about 5.6% of missing values, and *Native country* has about 1.7% missing values. The aggregation plot (right) shows 5.5% of samples are missing values for both *Work class* and *Occupation*. Less than 2% of samples are missing just *Native country* and less than 1% are missing all three features.

Figure 2 shows the frequency of observed categories and missing values for *Work class* and *Occupation*. Each stacked column shows the proportion of missing values in the other feature and *Native country* for each category. The plot shows the missing values are not MCAR: individuals working in the private sector, for instance, are more likely to have missing values than those individuals in other work classes. However, missing values tend to be evenly distributed across occupational categories.

3.3. Preprocessing

We preprocess the training and test data as follows:

*We note, however, that a property test can be as complex as the data at hand.

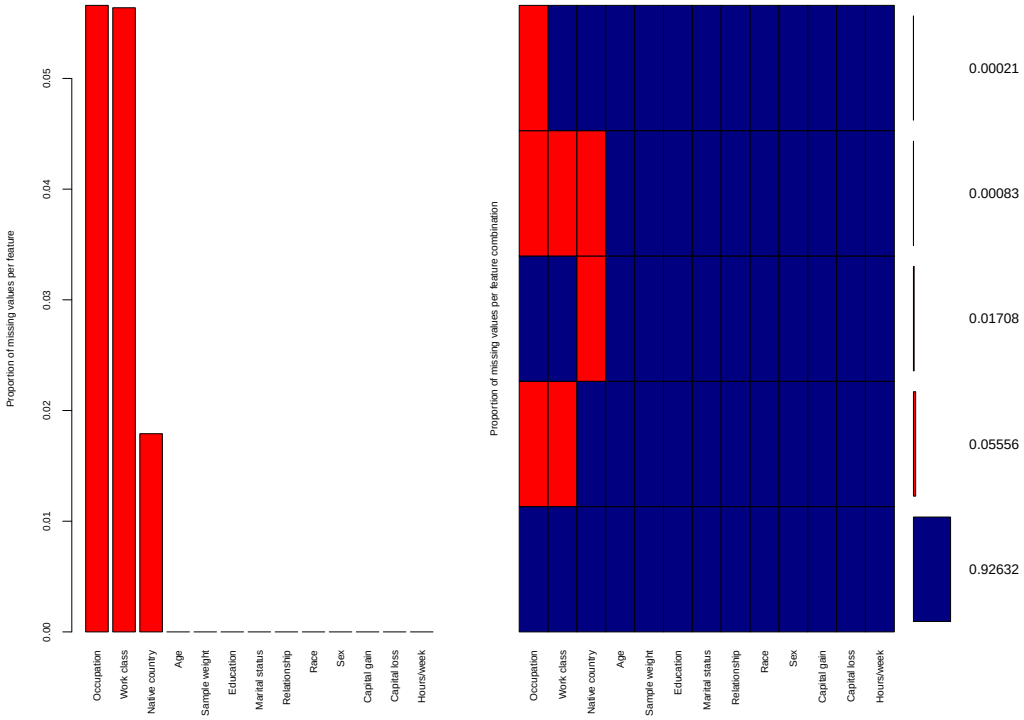

Figure 1. Histogram of proportion of missing values in each feature (Left) of Adult training set and aggregation plot of all existing combinations of missing and non-missing values in the samples (Right).

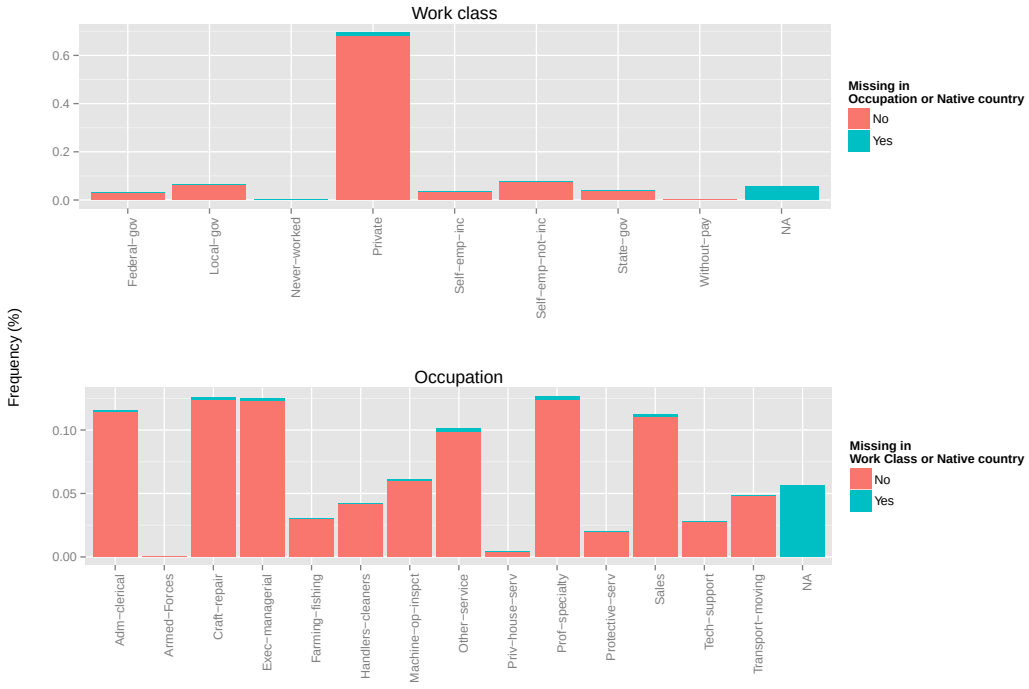

Figure 2. Barplot of proportion of observed and missing values of *Work class* and *Occupation* in Adult dataset.

1. Drop the string feature `education` because it contains identical information as its numeric counterpart `education.num`.

2. Binarize the categorical variables.

3. Implement imputation technique. When replacing the missing data with the mean, median, or mode of the feature vector, the summary statistic is computed from the training data, not from the test data.

4. Standardize each feature to midrange 0 and range 2 (i.e., minimum -1 and maximum 1). We use the training set values for range and midrange to standardize test set features.

3.4. Model selection

We implement two neural network classifiers: a simple model and a complex model. The complex model uses two hidden layers, both with the same size of hidden nodes, which is computed using the formula $N_h = \frac{N_s}{(\alpha * (N_i + N_o)}$, where the number of hidden nodes N_h is a function of the number of training examples N_s, a scaling parameter α, the number of input neurons N_i, and the number of output neurons N_s. The complex model uses root-mean-square gradient scaling for the update rule, while the simple model uses one hidden layer and stochastic gradient descent for the update rule. Both models use a cross–entropy cost function and weights initialized randomly using Gaussian distribution and scaled by a factor of 10^{-2}.

We use two layers of cross–validation for selecting model parameters: grid search and K–fold cross validation. We perform an exhaustive search on a grid of parameter values composed of $\alpha = \{1, 4, 9\}$ learning rate $\gamma = \{10^{-1}, 10^{-2}, 10^{-3}\}$, and $\text{mini} - \text{batch size} = \{32, 512, 4096\}$. For each $3! * 3 = 18$ parameter combinations, we perform K–fold cross–validation on $k = 3$ folds. We evaluate the models based on the average error rate across folds and also record average cost and average computing time across folds.

We train both simple and complex classifiers on data with missing values imputed using most of the techniques described in Section 2.1 and on data with instances with missing values removed. Figure 3 provides a visual example of the results of the simple neural network on training data with missing values imputed using random forests. Table 2 reports the models with the lowest cross–validated error rate. The simple neural network outperforms the complex classifier in most cases. We find that omitting instances with missing values yields a lower error rate on the training data than all of the imputation methods. The difference between imputation methods in terms of accuracy is minimal.

3.5. Model assessment

We select the three highlighted models in Table 2 to train on the entire training set and then fit each model on the test features. We handle missing values in the test features in the same manner as the training features. The simple neural network trained on data with missing values imputed using random forests yields the highest accuracy (16.63% error rate), followed by mode imputation (16.84%), and missing values removed (18.59%).

4. Conclusion

Neural networks have become a popular machine learning algorithm in many domains, in part due to the ability of neural networks to "learn" how to engineer features. However, researchers analyzing survey data typically choose decision trees or random forests for prediction tasks because missing data and categorical variables are not easy to handle with neural networks. This paper investigates techniques for handling missing data and encoding categorical data such that it is appropriate to neural network classifiers. We compare six different imputation strategies, a simple and complex neural network classifier, and a 3x3 parameter grid.

We use the Adult dataset for benchmarking because it has a mixture of continuous and categorical variables and has under 10% of the instances containing missing values. Removing instances with missing values instead of imputing the missing values actually yields the highest cross–validated accuracy on the training set. This finding suggest that the instances with missing values or the features that contain missing values (i.e., *occupation*, *work class*, and *native country*) do not contribute meaningful information for the prediction task. Another interpretation is that there is not enough missing values in the training data for imputation to make a difference.

However, we find that two of the imputation methods — prediction using random forests and replacing the missing values with column modes — both yield lower generalization error than no imputation. Our lowest test error is within 3% of the state–of–the–art, which uses Naive Bayes and removes data with missing values.

For future work, we will explore the relevance of the features with missing data and the amount of missing data for this particular prediction task.

Imputation method	Model type	α	γ	Batch size	Error rate	Cost	Time (min.)
Remove instances with missing values	Simple	9	0.1	32	0.1406	0.1525	3.7
Factor analysis (PCA)	Simple	4	0.1	32	0.1411	0.2487	8.3667
Prediction model (random forests)	Simple	4	0.1	32	0.1413	0.2056	6.7
Summary statistic (mode)	Simple	9	0.1	32	0.1415	0.1829	4.2667
Case substitution	Simple	1	0.1	32	0.1417	0.2189	6.8
Prediction model (random forests)	Complex	1	0.1	32	0.1426	0.2919	9.9
Factor analysis (PCA)	Complex	1	0.01	32	0.143	0.3175	11
Remove instances with missing values	Complex	1	0.1	32	0.1439	0.1778	8.5
Summary statistic (median)	Simple	4	0.1	32	0.1446	0.1924	-
Summary statistic (mean)	Simple	4	0.1	32	0.1453	0.1915	-
Summary statistic (mean)	Complex	1	0.1	32	0.1593	0.3118	-
Summary statistic (median)	Complex	1	0.01	32	0.1595	0.3564	-

Table 2. Performance of models selected on the basis of cross–validated error rate on the training data. `Imputation method`is how missing values in the training data are imputed; `Model type` is the type of neural network classifier used; α is the scaling factor used to determine the number of hidden neurons in the neural network; γ is the learning rate; `Batch size` is the size of the batch; `Error rate` is the mean 3-fold cross–validated error rate on the training data; `Cost` is the mean cross–entropy cost across folds; `Time` is the mean training time across folds.

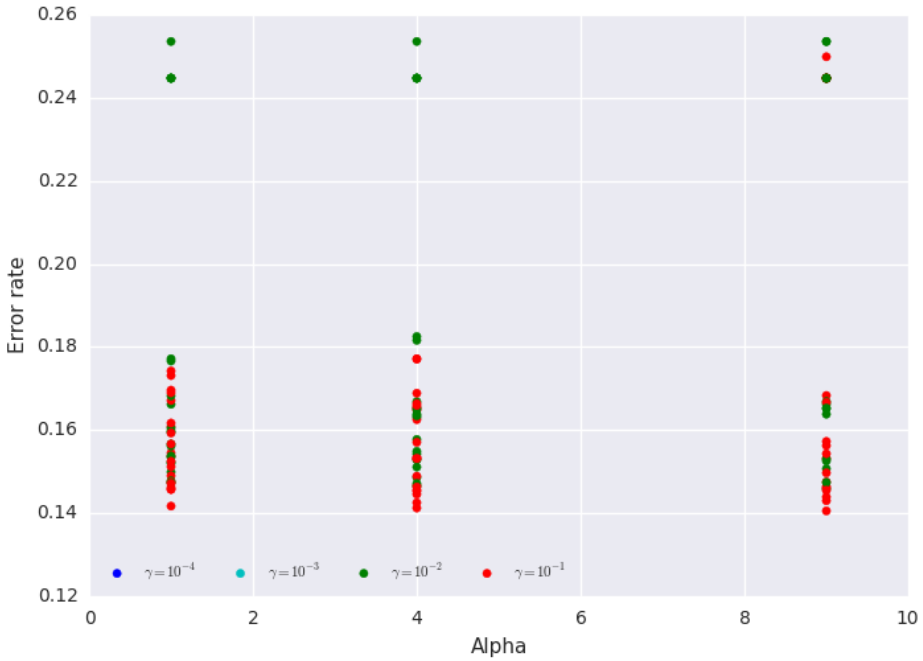

Figure 3. Performance of simple neural network on training data with missing values imputed using random forests: 3-fold cross–validated error rate versus α (x–axis) and γ (colors). See Table 2 for definitions.

References

[1] G. E. Batista and M. C. Monard. An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, 17(5-6):519–533, 2003.

[2] R. K. Brouwer. A feed-forward network for input that is both categorical and quantitative. *Neural Networks*, 15(7):881–890, 2002.

[3] E. D. De Leeuw, J. Hox, M. Huisman, et al. Prevention and treatment of item nonresponse. *Journal of Official Statistics-Stockholm*, 19(2):153–176, 2003.

[4] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37, 1996.

[5] C.-C. Hsu. Generalizing self-organizing map for cate-

gorical data. *Neural Networks, IEEE Transactions on*, 17(2):294–304, 2006.

[6] R. Kohavi. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *KDD*, pages 202–207. Citeseer, 1996.

[7] M. Lichman. UCI machine learning repository, 2013.

[8] NAPP. Minnesota population center. north atlantic population project: Complete count microdata. version 2.0 [machine-readable database]. *Minneapolis, MN: Minnesota Population Center, available at* `https://www.nappdata.org`, 2008.

[9] S. Ruggles, T. Alexander, K. Genadek, R. Goeken, M. Schroeder, and M. Sobek. Integrated public use microdata series (ipums): Version 5.0 [machine-readable database]. *University of Minnesota, Minneapolis, available at* `http://usa.ipums.org`, 2010.

[10] H. Wang, G. Xing, and K. Chen. Categorical data transformation methods for neural networks. In *IKE*, pages 262–266, 2008.