Investigations into Missing Values Imputation Using Random Forests for Semi-supervised Data

Tsunenori Ishioka
The National Center for University Entrance Examinations
2-19-23 Komaba, Meguro-ku
Tokyo, Japan
tunenori@rd.dnc.ac.jp

ABSTRACT

This paper presents a revised procedure that imputes missing values by using random forests on semi-supervised data. The method has a feature that not only allows missing data to be found in a response variable but in a predictive variable, and furthermore, it can now deal with any types of data, i.e., numerical values, categories and categories with an order. By evaluating this method using Titanic data and eleven UC Irvine repository datasets, we found that our method performed fairly well, and a method of naive median imputation was also suitable in these cases.

Categories and Subject Descriptors

G.4 [**Mathematical Software**]: Algorithm design and analysis; I.2.6 [**Artificial Intelligence**]: Learning—*analogies*

General Terms

Algorithms

Keywords

Ensemble learning, data imputation, missing data, R, rfImpute, UCI machine learning repository

1. INTRODUCTION

We often see that missing values are included when analyzing real-world data. Some information, especially in wireless environments, tends to get lost. The easiest way of treating missing values is to "remove" them. Another way is to make a reasonable inference from the observations to account for the missing data. The missingness, i.e., the randomness of missing data, can be divided into three classes [14]: (1) missing completely at random (MCAR), (2) missing at random (MAR), and (3) not missing at random (NMAR). We often use the full information maximum likelihood method (FIML) [6] in the case of (1) or (2). We usually introduce auxiliary variables in the case of (3), and can get the model closer to an assumption of MAR.

If labeled or response variable data are missing in the issue of class classification or regression, the dataset is called "semi-supervised data." Since the unlabeled data are relatively easy to obtain, techniques that make use of unlabeled data for training — typically a small amount of labeled data with a large amount of unlabeled data, have the potential to produce accurate estimates.

The predictor as well as response variable naturally include missing values. Note that FIML cannot handle semi-supervised data that include missing values in any predictors; it can only treat supervised data that include missing predictors. On the contrary, many semi-supervised learning (SSL) methods, such as low-density separation approaches [4] and graph-based methods [16] and so on [18], only allow responses missingness under natural conditions (Table 1).

Here, we will impute these missing values by using random forests. Random forests [1] is well-known as a substantial modification of bagging techniques, and is scalable for high dimensional data. It builds a large collection of decorrelated trees and then averages them. It is mainly used as an accurate classifier or regression tree. The latest Fortran77 code by Breiman and Cutler [3] is Version 5.1, dated 2004. Version 4 contains modifications and major additions to Version 3.3, and it allows missing predictor values to be replaced through two options [2]. The first is "missquick" (Ver. 4), which replaces all missing values by the median of the non-missing values in their column, if they are real, or by the most numerous value in their column if they are categorical. The second is "missright" (Ver. 5). This option starts with "missquick" but then iterates by using proximities and does an effective replacement even with a large amount of missing data. Missing values are represented by a proximity weighted sum over the non-missing values.

Liaw implemented these ideas in a statistical environment R [13], calling them "na.roughfix" and "rfImpute" [12]. These R functions unfortunately cannot be used in semi-supervised cases [5]. Therefore, we extended their ideas to semi-supervised data [11], and presented the procedure in R. This starts with the rough imputation of the missing data ("na.roughfix") and repeatedly obtains new proximities of the semi-supervised data by running random forests.

Table 1 summarizes the availability of various methods whose variables allow their missing values. FIML [6], which is often shortened as the maximum likelihood method (ML), only allows predictors missingness. SSL methods[4, 16, 18] are completely opposite to FIML. Random forests [1] only

iiWAS 2014, 4-6 December, 2014, Hanoi, Vietnam.

Table 1: Should missing data be allowed?

Methods	Predictors	Response var.
FIML(ML) [6]	Available	Not available
SSL methods[4, 16, 18]	Not available	Available
Random forests [1]	Not available	Not available
Random forests [3]	Available	Not available
RfImpute [12]	Available	Not available
Our procedure [11]	Available	Available

operates in complete data. The latest random forests [3] and rfImpute [12] can impute the missing predictors. In all, only our method [11] can handle the missing data in any of the predictors and response variables, and it operates normally.

We found in our previous research [11] that our method had higher rates of correct classification using a spam dataset collected at Howlett-Packard Labs, and Edgar Anderson's iris data. The former dataset classifies 4601 e-mails as spam or non-spam, and has 58 variables indicating the frequency of certain words and characters in the e-mail. The latter provides measurements in centimeters of the variables sepal length and width and petal length and width, for 50 flowers from each of three species of iris. The species are "iris setosa," "versicolor," and "virginica". That is, the iris dataset has 150 observations and five variables. Both datasets are typical test cases for many classification techniques in machine learning.

However, neither of these two datasets includes intrinsic missing values. The missing values were created by dropping their original values artificially in order to evaluate the classification. In addition, all the predictor or explanatory variables of these datasets are numeric. The predictors in some cases contain categorical and/or ordered categorical variables besides numeric ones. Therefore, we extended our procedure so that the variables of any types could be dealt with.

In this research, we evaluate the properly imputing missing values of intrinsic missing datasets stored at the UC Irvine (UCI) Repository of Machine Learning Databases. Section 2 summarizes the elements of a technique that imputes the missing values for semi-supervised data. Section 3 explains the revised procedure for imputing them. Section 4 presents the results obtained with the Titanic dataset, and 11 examples stored at the UCI Repository. All datasets contain intrinsic missing values. Section 5 concludes the paper.

2. RFIMPUTE

2.1 Proximity measure

The (i, j) element of the proximity matrix produced by a random forest is the fraction of trees in which elements i and j fall on the same terminal node. "Similar" observations should intuitively be on the same terminal nodes more often than dissimilar ones. The proximity matrix can be used to identify structures in the data and used for semi-supervised learning with random forests.

When treating random forests, we should note that each tree uses a part of input variables. Suppose L represents input variables, and number $\ell \ll L$ is specified such that at each node, ℓ variables are selected at random out of L. Since the growth of the tree does not depend on the number of cases but on the number of variables (ℓ), no tree will increase in size. Therefore, the probability of falling on the same terminal node will not become very small. It is appropriate to use this as an index of the degree of proximity.

2.2 R procedure

Missing values are indicated by 'NA's in R [13]. A function returning a result of random forests is the "randomForest," developed by Liaw [12]. The algorithm starts by imputing NAs by using "na.roughfix." Then, "randomForest" is called with the completed data. The proximity matrix output from the "randomForest" is used to update the imputation of the NAs. The imputed value for continuous predictors is the weighted average of the non-missing observations, where the weights are the proximities. The imputed value for categorical predictors, is the category with the largest average proximity. This process is iterated a few times.

3. REVISED PROCEDURE

3.1 Missing value replacement for training set

Our procedure, as well as Liaw's "rfImpute," involves two ways of replacing missing values. The first way is fast. If the mth variable is not categorical, the method computes the median of all values of this variable in class j; then it uses this value to replace all missing values of the mth variable in class j. If the mth variable is categorical, the replacement is the most frequent non-missing value in class j. These missing values are replaced or filled by "na.roughfix."

The second way of replacing missing values is computationally more expensive but outperforms the first, even with large amounts of missing data. It begins by roughly and inaccurately filling in the missing values. The key technique is not to estimate the missing values on the basis of all proximities but the non-missing proximities. Then, it runs a forest procedure and computes the proximities.

If $x(n, m)$ is a missing continuous value, we estimate its fill as an average over the non-missing values of the mth variables weighted by the proximities between the nth case and the other cases. If it is a missing categorical variable, we replace it by using the most frequent non-missing value, where the frequency is weighted by proximity.

In summary, when there is a missing continuous value, we use

$$\widehat{x}(n,m) = \frac{\sum_{i \neq n; i \in \text{non-missing}} \text{prox}(i,n) x(i,m)}{\sum_{i \neq n; i \in \text{non-missing}} \text{prox}(i,n)}, \qquad (1)$$

where $\text{prox}(\cdot,\cdot)$ is the proximity.

When there is a missing categorical variable, we use

$$\widehat{x}(n,m) = \underset{C_m}{\text{argmax}} \sum_{\substack{i \neq n \\ i \in \text{non-missing}}} \text{prox}(i,n), \qquad (2)$$

where C_m means mth categorical variables.

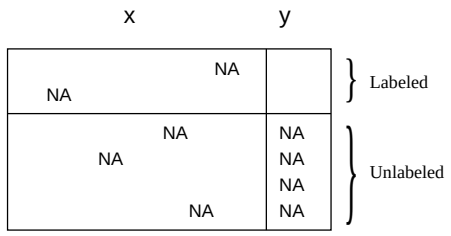
Now, it is possible to iteratively construct a forest again by using these newly filled-in values, find new fills, and reiterate. However we cannot do this. We found through experience that the iterations did not improve performance.

The main reason we only use non-missing data in (1) is that the imputation of missing data is not stable. Even if the proximities to the target are rather close to one (=1),

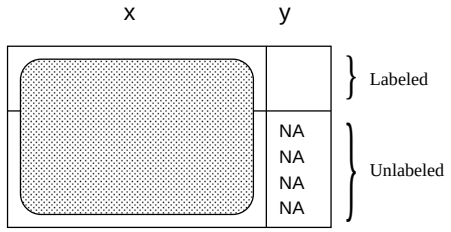
the proximity associated with missing data would not been reliable. Our numerical investigations revealed that our procedure using non-missing data is better than using all data. The same thing occurred in (2).

3.2 Algorithm

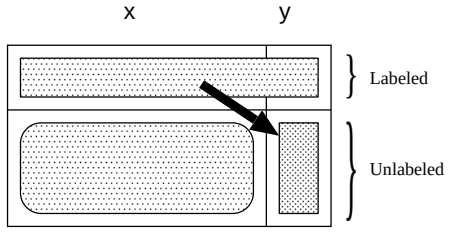
Semi-supervised learning ought to treat missing response variable (y) as a training data set. Since our method is to immediately impute missing data, both predictor variables (x) and response variable (y) can include missing values (Fig.1: Step 0). Semi-supervised learning, which includes large amounts of missing predictor variables (x) and missing response variables (y), has the potential to considerably cover real world data. A good method of semi-supervised learning yields many benefits. Our procedure is outlined in Figure 1.



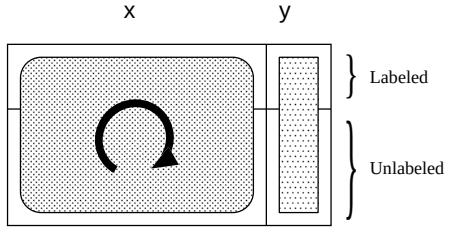
(Step 0) Initial state



(Step 1a) Imputation of predictor variables (x)



(Step 1b) Imputation of response variable (y) based on labeled data



(Step 2 and 3) Replace missing predictor variable (x)

Figure 1: Procedure for missing (NA) data imputation

(Step 0): There are labeled data and unlabeled data; NA indicates missing data.

(Step 1): By starting with a rough imputation of missing predictor variables (x), we estimate the missing response variable ($\hat{y}$) by running the random forests algorithm.

(Step 2): We replace the missing predictor variables ($\hat{x}$) by using the proximities between cases and estimate the response variable ($\hat{y}$).

(Step 3): If the imputed values ($\hat{x}$) converge, we output them ($\hat{x}, \hat{y}$). Otherwise, we repeat Step 2.

We call this procedure "rfImput.smsupvsd," which means "an imputation method using random forests for semi-supervised learning." We found that Step 3 did not produce significant improvements. The procedure has been developed by our previous work [11]. Minor changes to the program are made for ill conditions.

3.3 R Tips to handle any types of data

A program that calculates the spatial distance between two vectors to any data types (numeric, categorical, and ordered categorical) needed to be created to apply our method to all data types.

The program code in R is described below:

```
# Description:
#   Return relative distance between 'x.impute' to 'x.org'
# Arguments:
#   x.impute: imputed data
#   x.org: original data
dist.rel <- function (x.impute, x.org){
 ncol.x <- length(x.org)
 x.abs.org <-  matrix(abs(as.numeric(unlist(x.org))), ncol=ncol.x)
 max.x <- apply(x.abs.org, 2, max) # normalize the features size
 # 'x.impute' and 'x.org' may include factor elements
 if (FALSE){ # available for only numeric
  diff.x <- (x.impute - x.org) / max.x # normalize
  diff.rel <- sum(diff.x^2) / sum((x.org / max.x)^2)
 }else{
  mat.x.impute <- matrix(as.numeric(unlist(x.impute)), ncol=ncol.x)
  mat.x.org <- matrix(as.numeric(unlist(x.org)), ncol=ncol.x)
  max.numx <- as.numeric(unlist(max.x))
  diff.x <- sweep((mat.x.impute - mat.x.org), 2, max.numx, FUN="/")
  size.org <- sweep(mat.x.org, 2, max.numx, FUN="/")
  diff.rel <- sum(diff.x^2) / sum(size.org^2)
 }
 return(diff.rel)
}
```

The former code is saved at the portion of "`if (FALSE){ }`," which was effective for the numerical data type. Now, the portion by "`else{ }`" operates and fulfills the conditions we required. The key technique is transforming any types of objects to numeric values using functions of "`unlist`" and "`as.numeric`."

Other tips are reading missing values indicated by 'NA' or '?', not as a character string but as the missing data of R. For that purpose, we read the given data as follows.

```
x <- read.csv(filename, na.strings=c("?","","NA"))
```

Data types other than those missing are held as they are by using this.

All related R programs can be viewed at `http://www.rd.dnc.ac.jp/~tunenori/rfImpute.html` .

4. APPLICATION EXAMPLES

4.1 Titanic3 Datasets indicating survival status

Thomas Cason of the University of Virginia has greatly updated and improved the 'titanic' data frame using the Encyclopedia Titanica and created a new dataset called 'titanic3.' This data frame describes the survival status of individual passengers on the Titanic [9]. It is available at [10].

The data frame has 1309 observations on the following 14 variables:

pclass A factor with levels 1st, 2nd, and 3rd
survived Survival (0 = No; 1 = Yes)
name Name
sex A factor with levels female and male
age Age in years
sibsp Number of siblings/spouses aboard
parch Number of parents/children aboard
ticket Ticket number
fare Passenger fare
cabin Cabin
embarked A factor with levels Cherbourg, Queenstown, and Southampton
boat Lifeboat
body Body identification number
home.dest Home/Destination

As the **boat** number immediately indicated whether someone had survived or not, it was removed from the object of analysis. Moreover, since these four of **name**, **ticket**, **body**, and **home.dest** (home/destination), do not involve whether someone had survived or not, we did not use these.

We totally used **survived** as a response variable and eight remaining variables, i.e., **pclass**, **sex**, **age**, **sibsp**, **parch**, **fare**, **cabin**, **embarked**, as predictor variables.

Now, we must be cautious of two things here.

1. There are three types of predictor variables: numeric, categorical and ordered categorical variables. **Age** and **fare** are numeric. **Sex**, **cabin**, and **embarked** are categorical. **Pclass**, **sibsp**, and **parch** are ordered categorical, which are indicated as integers.

   Random forests can deal with these types of data, i.e., a mixture of numerical and categorical data, without special treatment.

2. Missingness does not occur completely at random. While the total average of the missing ratio is 10.9%, the missing ratio of **cabin** is 77.5% (1014/1309). The ratio of passengers expressed as a percentage without missingness is 79.4% (1039/1309). That is, missingness is inclined toward a specific variable.

To evaluate the performance of our "rfImput.smsupvsd," we compared it with two methods.

1. Median Imputation: "RandomForest" does not work for any y that includes missing responses. Hence, we filled missing x by replacing the column median, and configured the forest model for non-missing response cases (y) by using "RandomForest." Using this model, we estimated the response values ($\hat{y}$) for their missing y.

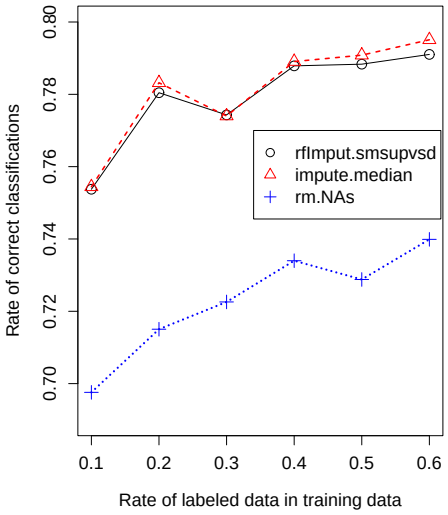


Figure 2: Correct classification for semi-supervised titanic data

2. Missing data elimination: We started to eliminate the data containing any missing x and missing y. Using this complete data, we built the "RandomForest" model. Therefore, no unlabeled data were used.

Semi-supervised as well as supervised learning predict or estimate y from x. Therefore, we used precision, i.e., the rate of correct classifications, as a criterion to evaluate the performance of learners.

We conducted 150 runs of 10-fold stratified cross-validation in this dataset. That is, 10% (= p) of the original data were put aside as the testing set to evaluate the performance of the learning algorithm. The remaining 90% (= $1 - p$) of data were divided into labeled and unlabeled data, according to a pre-defined percentage of labeled data (= q). This data-splitting setting is very common [8]. The q was set in our experiments, to be from 10% to 60%. Therefore, when $q = 0.2$, 10% of the data were kept as the testing set, 20% of the 90% data were randomly sampled as labeled data, while the remaining 80% of the 90% data were saved as unlabeled data.

Figure 2 plots the results for the three methods. A larger value on the vertical axis indicates better performance. A value of one (=1) means that all missing y have correctly been predicted. The horizontal axis shows the rate of the labeled data (q).

Note that the performance of the method of eliminating missing data is overestimated, because this method uses just complete data to evaluate performance, while the other two methods' use testing data containing the original missing data.

In general, the higher the training data rate on the horizontal axis is, the higher the value on the vertical axis becomes. Because the missing data are randomized, the lines on the graph do not always increase monotonously. Our method ("rfImput.smsupvsd") and median imputation method ("impute.median) are excellent on the same slope. The missing elimination method ("rm.NAs") is the worst because observations to build the model are fairly small.

Table 2: UCI Datasets including missing data

Name	% Missing	# Classes	Attribute types	# Instances	# Attributes	Year
Pittsburgh bridges	5.94	6	Categorical, integer	108	13	1990
Credit approval	0.61	2	Categorical, integer, real	690	15	
Cylinder bands	5.00	2	Categorical, integer, real	512	39	1995
Dermatology	0.06	6	Categorical, integer	366	33	1998
Echocardiogram	7.69	2	Categorical, integer, real	132	12	1989
Hepatitis	5.39	2	Categorical, integer, real	155	19	1988
Horse colic	13.81	2	Categorical, integer, real	368	27	1989
Lung cancer	0.27	3	Integer	32	56	1992
Mushroom	1.32	3	Categorical	8124	22	1987
Post-operative patient	0.37	2	Categorical, integer	90	8	1993
Soybean (Large)	6.44	19	Categorical	307	35	1988

4.2 UCI Repository Datasets

The next examples are well-known UCI Repository datasets [15], which are often used in the field of machine learning. We used all datasets whose size is not huge and that contains missing data. Eleven datasets correspond to the following states: eight binary class datasets, and the three multiclass datasets, all data types are multivariate, and all default tasks are for classification. The number of instances is from 32 to 8124, and the attributes is from 8 to 39. The missing rates are quite small. Table 2 summarizes their attribute information.

We conducted 50 runs of 10-fold cross-validation on each dataset, which means $p = 0.1$. Then, missing values were replaced into the training data by setting $q = 0.25$ as a percentage of the labeled data. The value of q was small to set up because typical semi-supervised learning adds a large amount of unlabeled data to a small amount of labeled data for training.

Table 3 lists the results in the ratio of correct classifications for each dataset. The three methods described in subsection 4.1 are compared. The asterisks '*' represent "impute.median" and/or "rfImput.smsupvsd" win "rm.NAs" on a dataset under a pair-wise t-test with a significance level of 95%.

The method of removing data containing missing values ("rm.NAs") is generally inferior to the other two methods. This is because of the small sample size of the training data. Moreover, our method ("rfImput.smsupvsd") is not always the best of the three, despite careful consideration. Our method was best of the three in previous research [11], using a spam dataset [15] collected at Hoewlett-Packard Labs and Anderson's iris dataset. Where the missing values are artificially produced; the values were randomly dropped.

Our method worked well in these previous situations, such as missing completely at random (MCAR), or missing at random (MAR), However, it did not in a situation of not missing at random (NMAR). In USI repository datasets at least, missingness tends to incline toward specific variables. A data-frame and the head portion of "Pittsburgh bridges" are listed below:

```
> str(x)
'data.frame':   108 obs. of  12 variables:
$ V2 : Factor w/ 4 levels "A","M","O","Y": 2 1 1 1 2 1 1 2 1 1 ...
$ V3 : num  3 25 39 29 23 27 28 3 39 29 ...
$ V4 : int  1818 1819 1829 1837 1838 1840 1844 1846 1848 1851 ...
$ V5 : Factor w/ 4 levels "AQUEDUCT","HIGHWAY",..: 2 2 1 2 2 2 1 2 1 2 ...
```

Table 3: Ratio of correct classification for semi-supervised datasets

Dataset name	rm.NAs	impute. median	rfImput. unsupvsd
Pittsburgh bridges	0.496	0.581*	0.588*
Credit approval	0.860	0.853	0.853
Cylinder bands	0.693	0.742*	0.740*
Dermatology	0.960	0.964	0.965
Echocardiogram	0.903	0.934*	0.935*
Hepatitis	0.855	0.835	0.836
Horse colic	0.685	0.802*	0.764*
Lung cancer	0.431	0.432	0.432
Mushroom	1.000	1.000	1.000
Post-operative patient	0.622	0.626	0.626
Soybean (Large)	0.786	0.782	0.709

```
$ V6 : int  NA 1037 NA 1000 NA 990 1000 1500 NA 1000 ...
$ V7 : int  2 2 1 2 2 2 1 2 1 2 ...
$ V8 : Factor w/ 2 levels "G","N": 2 2 2 2 2 2 2 2 2 2 ...
$ V9 : Factor w/ 2 levels "DECK","THROUGH": 2 2 2 2 2 2 2 2 1 2 ...
$ V10: Factor w/ 3 levels "IRON","STEEL",..: 3 3 3 3 3 3 1 1 3 3 ...
$ V11: Factor w/ 3 levels "LONG","MEDIUM",..: 3 3 NA 3 NA 2 3 3 NA 2 ...
$ V12: Factor w/ 3 levels "F","S","S-F": 2 2 2 2 2 2 2 2 2 2 ...
$ V13: Factor w/ 7 levels "ARCH","CANTILEV",..: 7 7 7 7 7 7 6 6 7 7 ...
```

At first glance, we find that attributes "V6" and "V11" include many missing values indicated as NA. Actually, "V6" contains 27 NAs of 108 instances; "V11" contains 16 NAs. As the average missing rate is about 5%, these values are quite large.

Moreover, some instances contain many NAs as follows.

```
> x[107:108,]
    V2   V3   V4       V5  V6 V7  V8    V9   V10   V11  V12    V13
107  O 43.0 1982  HIGHWAY  NA NA   G  <NA>  <NA>  <NA>    F   <NA>
108  A 28.0 1986  HIGHWAY  NA NA   G  <NA>  <NA>  <NA>    F   <NA>
```

Such tendencies are also generally the same in other datasets. Relatively naive "impute.median" succeeds in these environments or circumstances. We are disappointed that the UCI machine learning repository do not contain typical semi-supervised datasets that consist of a few labeled data with a lot of unlabeled data.

5. CONCLUSIONS

The procedure was extended so that our method could be applied to all data types. Further, its validity was evaluated by choosing datasets that contained missing values from the UCI repository and Titanic data.

Since our method imputed missing x in unlabeled data, a random forests (RF) model for unlabeled data could also be built. Therefore, it was possible to use both RF models together for labeled and labeled data, and a better integrated RF model by using the two could be built. However, this model did not work well. No matter how it might adjust the weights of two models, this integrated RF model was not able to outperform the single RF model for labeled data.

Since $\widehat{y}$ in unlabeled data is a primal estimate, $\widehat{x}$ based on this does not create improvements. Only the effect on $\widehat{x}$ that spreads the error of $\widehat{y}$ is caused.

If we consider the omnipresent nature of missingness, i.e., a specific variable has many NAs, and/or a specific instance has a tendency with many NAs, it may be effective to weight $\widehat{y}$ according to the number of complemented NA(s). Another method of co-training [17] that applies two basic learners to train the data source, which uses the most confident unlabeled data to augment labeled data in the learning process, may be also effective. We would like to tackle this task in the future.

Acknowledgments

The author is grateful to three anonymous reviewers for their constructive comments. This work was supported by Grant-in-Aid for Scientific Research No. 26350357 and 25350311.

6. REFERENCES

[1] L. Breiman, Random forests, *Machine Learning*, **45** (1), 5–32, 2001.

[2] L. Breiman, *Manual for Setting Up, Using, and Understanding Random Forest V4.0*, `http://oz.berkeley.edu/users/breiman/Using_random_forests_v4.0.pdf`, 2003.

[3] L. Breiman and A. Cutler, Random forests, `http://www.stat.berkeley.edu/~breiman/RandomForests/` updated March 3, 2004.

[4] O. Chapelle and A. Zien, Semi-supervised classification by low density separation, In Proceedings of the Tenth International Workshop on *Artificial Intelligence and Statistics*, 57–64, 2005.

[5] CRAN, *Package randomForest*, `http://cran.r-project.org/web/packages/randomForest/randomForest.pdf`

[6] C. K. Enders, A note on the use of missing auxiliary variables in FIML-based structural equation models. *Structural Equation Modeling: A Multidisciplinary Journal*, 15, 434-448, 2008.

[7] A. Gelman and J. Hill, *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Cambridge University Press, 2007.

[8] Y. Guo, H. Zhang, and X. Liu. Instance selection in semi-supervised learning. In proceeding of *Advances in Artificial Intelligence* – 24th Canadian Conference on Artificial Intelligence (Canadian AI 2011), Canada, 2011.

[9] F. E. Harrell Jr., *Regression Modeling Strategies with Applications to Linear Models, Logistic Regression, and Survival Analysis*. Springer. 2001.

[10] F. E. Harrell Jr., Titanic Data, `http://www.stats4stem.org/r-titanic3-data.html`, 2002.

[11] T. Ishioka, Imputation of missing values for semi-supervised data using the proximity in Random Forests, *iiWAS 2012*, Bali, 309-312, 2012.

[12] A. Liaw, Missing value imputations by randomForest, *R Documentation*, `http://www.stat.ucl.ac.be/ISdidactique/Rhelp/library/randomForest/html/rfImpute.html`

[13] The R Project for Statistical Computing, `http://www.r-project.org/`

[14] D. B. Rubin, *Multiple imputation for nonresponse in surveys*, New York: Wiley, 1987.

[15] UCI Repository of Machine Learning Databases, `http://www.ics.uci.edu/~mlearn/MLRepository.html`

[16] G. C. Valls, T. M. Bandos, and D. Zhou, Semi-supervised graph-based hyperspectral image classification, *IEEE Transactions on Geoscience and Remote Sensing,* **45** (10), 3044–3054, 2007.

[17] J. Xu, H. He, and H. Man, DCPE Co-training for classification, *Neurocomputing*, **86**, 75–85, 2012.

[18] X. Zhu, Semi-supervised learning literature survey, TR-1530, University of Wisconsin-Madison Department of Computer Science, 2005 (last modified on July 17, 2008).