Impute vs. Ignore: Missing Values for Prediction

Qianyu Zhang, Ashfaqur Rahman, and Claire D’Este

Abstract—Sensor faults or communication errors can cause certain sensor readings to become unavailable for prediction purposes. In this paper we evaluate the performance of imputation techniques and techniques that ignore the missing values, in scenarios: (i) when values are missing only during prediction phase, and (ii) when values are missing during both the induction and prediction phase. We also investigated the influence of different scales of missingness on the performance of these treatments. The results can be used as a guideline to facilitate the choice of different missing value treatments under different circumstances.

I. INTRODUCTION

Event detection from multiple environmental sensor data streams can add value to real-time decision support systems for industries such as agriculture and aquaculture [1]. However, sensors placed in the field commonly experience faults and communication errors. This can cause missing values in the data that we require both for training our classifiers and for gaining a classification from them. Frequently failing to provide a classification will result in a reduction of trust in the overall decision support system. This trust will also be based on the accuracy of the classifications, which means we do not want to throw away training examples if we can avoid it. Therefore we require a means for training and providing a classification even when some of the features are not available.

Missing value treatments in machine learning are a solution to this problem. There are two alternative approaches to deal with missing values: Impute and Ignore. Imputation treatments [2][3] fill in attributes in the instance vector using statistical techniques and the complete vector is fed to the predictor. Ignore treatments (also called reduced model or ensemble classifier treatments) [4][5] overlook the missing attributes, produce a vector based on the available attributes, and feed that vector to a predictor trained on those particular attributes. In current literature, imputation treatments have been widely investigated [6][7], but ignore treatments draws less attention.

Ignore and impute treatments have been previously compared on data sets that are naturally occurring with missing values [5][8]. There are also studies that compare the performance of various missing value treatments under various scales of missingness [8][9][10][11]. Farhangfar [9] compared imputation methods under different scales of missingness. Pelckmans [10] studied supported vector machine classifiers tested on a modified version of the Ripley dataset. Melville [11] provides a comparison between the bagging [12], boosting [13] and DECORATE [14] methods. Concluding that DECORATE outperforms the others when features go missing, where DECORATE is an ensemble method that contains classifiers trained on both the original data and some artificial data. Garcia [8] compared most popular missing value treatments, using natural dataset with missing values and using dataset with various scales of missingness introduced to its most relevant feature.

We observe that the above studies are all restricted to certain circumstances and more or less focuses on methods involving artificial data. All such studies uses data with missing values into both the training and testing set, some only introduces missing values to a single feature in the data. Motivated by the above fact the research presented in this paper aims to compare the performance of the impute and ignore treatments under a broader and more practical set of situations, in order to provide true comparisons for potential applications. In detail, this paper investigates the performance of the impute and ignore treatments under different scales of missingness, (i) when values are missing only during prediction phase, and (ii) when values are missing during both the induction and prediction phase. For each instance, all features have an equivalent chance of being set missing. Moreover, the largest missing percentage considered in previous studies was 40%, we extended this to 50% in order to ensure the overall trend and to further support our results.

We have artificially introduced missingness at different scales on a number of benchmark data sets from the UCI machine learning repository [15] and analysed the performance of these treatments under different scenarios using the decision tree [16], nearest neighbour [17], and Bayesian network [18] classifiers. With different treatments and controlled introduction of missingness, the contribution in this paper can be considered as providing guidelines to facilitate the choice of missing value treatments under different circumstances.

The paper is organized as follows: some commonly used missing value treatments are presented in Section II. The experimental set-up and methodology is presented in Section III. Results are analysed in Section IV. Section V concludes the paper.

II. MISSING VALUE TREATMENTS

The most commonly used imputation treatments are mean, nearest neighbour, and regression [5]. The reduced model/ensemble classifier is the potential ignore treatment. The statistical approaches relies on the assumption that values are missing completely at random (MCAR) [19]. MCAR refers to the case where the probability of the value
TABLE I

Example used to demonstrate different missing value treatments

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>1</td>
</tr>
</tbody>
</table>

of an attribute going missing is independent of both observed and unobserved attribute values. Each of these treatments are detailed with the example presented in Table I:

1) **Impute**: Value imputation [2][20] replaces missing values with values estimated using existing data, this is most commonly used in the statistics community.
   - **Mean** imputation treatment replaces a missing entry with the statistical mean of the available values of corresponding attribute. In the example presented in Table I, we can replace all unknown values in attribute A by the mean of the known values in A, which is \((1+4)/2=2.5\).
   - **Nearest Neighbour** imputation treatment replaces missing values of an instance using the values of its nearest neighbour. The similarity between instances is determined using Euclidean distance, which is calculated based on the values of the available attributes.
     In the example presented in Table I, attribute A is missing in the middle instance. In terms of the available attribute B, the difference between the first and the middle instance is 4, whereas the difference is only 2 between the middle and the third instance. So the nearest neighbour is the third instance and the nearest neighbour imputation treatment will replace the missing value with 4.
   - **Regression** imputation treatment derives a relationship between the missing attribute and other available attributes based on the instances in the training data with the same missing pattern. More precisely, the attribute under consideration is expressed as a linear combination \[a_1x_1 + a_2x_2 + ... + a_nx_n\] where \(x_1, x_2, ..., x_n\) are the available attributes and the parameters \(a_1, a_2, ..., a_n\) are derived from the available instances.
     In the example presented in Table I, it is clear from instance one and three that A is equal to half of B. Applying this function, regression imputation estimates A to be \(0.5 \times 6 = 3\).

2) **Ignore**: These treatments ignores the missing values instead of replacing them with artificial values.
   - **Case deletion** treatment simply removes any training example for which we do not have a full set of attributes. This treatment risks losing important information from the remainder of the example at the training stage. It is also not possible to use this technique at the prediction stage. Therefore we do not present results in the paper for this treatment.
   - **Ensemble classifier** treatment alternatively trains specific classifiers for each possible missing pattern that might be present. During the training/induction phase, we first partition the data into different subsets according to their missing patterns. Each subset of data is then used to train a classifier, together the group of classifiers form the ensemble classifier. Each test instance is classified using the classifier trained on the group of training data with the same missing pattern. The procedure is illustrated in Fig. 1.

Fig. 1. Example of ensemble classifier treatment

The imputation treatments introduce artificial values leading to estimation error. This will be propagated or even exaggerated during classification. Whereas ignore treatments using the ensemble classifier eliminates this estimation error. However, one limitation is that this treatment requires larger memory for the storage of the multiple classifiers. An alternative could be to train the classifier for a particular missing pattern only when a test instance with such missing pattern is encountered.

III. EXPERIMENTAL SETUP

In this section we present the data sets used in the experiments, the process of introducing missing values, and two different experiment models: (i) when values are missing during prediction only and (ii) when values are missing during both induction and prediction.

A. Data Sets

We have used nine benchmark datasets from UCI machine learning repository [15], Table II provides a brief summary to the data sets. All the attributes in these data sets are numeric.
### TABLE II
**Summary of Data Sets**

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Instances</th>
<th>Attributes</th>
<th>Nominal Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast Cancer</td>
<td>683</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>Diabetes</td>
<td>768</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>Ecoli</td>
<td>336</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>Glass</td>
<td>218</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>Haberman</td>
<td>306</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Iris</td>
<td>150</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Liver</td>
<td>345</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>Thyroid</td>
<td>220</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Transfusion</td>
<td>748</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

### TABLE III
**Summary of Classifier Configurations**

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Parameter setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Tree</td>
<td>Confidence factor: 0.25</td>
</tr>
<tr>
<td></td>
<td>Minimum # of instances per leaf: 2</td>
</tr>
<tr>
<td></td>
<td>Number of folds used for pruning: 3</td>
</tr>
<tr>
<td>Bayesian Network</td>
<td>Estimator algorithm: simple estimator</td>
</tr>
<tr>
<td></td>
<td>alpha(initial count): 0.5</td>
</tr>
</tbody>
</table>

**B. Introduction of Missing Values**

We artificially introduced missing values into the data sets for the purpose of these experiments. We assume that all attributes are important for class prediction and hence do not perform a attribute selection stage. A spectrum of 0 to 50% missingness is considered. Zero missingness is when the full data set is being used. The performance of the classifiers on the full data set can be used as a benchmark for the use of various treatments and classifiers as missingness is introduced. When x% missingness is introduced, each attribute value has x% chance of being set missing, regardless of its value or position. More precisely x% attribute values are removed randomly. Note this way of introducing missingness results in a data set with no missing values in the class attribute. The resulting data set is also in accordance with the MCAR principle i.e. the probability of a value going missing is independent of both observed and unobserved attribute values.

**C. Classifiers**

We investigated the various treatments with a range of classifiers; namely the decision tree classifier, the nearest neighbour classifier, and the Bayesian network classifier. We employed the classifier implementation in WEKA [21], with parameters shown in Table III.

**D. Experiment Procedure: Values Missing in Test Set Only**

1) Randomize and stratify the data.
2) Split the data set into a training set containing the first 90% of instances, and the rest 10% forms the test set.
3) Introduce missing values into the test set.
4) Train and test the classifiers
   a) For the basic classifiers:
      i) Train the classifiers using the training data.
      ii) Impute the missing values in the test data.
      iii) Test the classifiers using the imputed test data.
      iv) Record the resulting classification accuracy.
   b) For the ensemble classifier treatment:
      i) Enumerate all possible missing patterns.
      ii) Sort the training data into groups according to the missing patterns. Note that since there are no missing values in this situation, all instances should be included in all the groups, with the corresponding attributes deleted.
      iii) Train a sub-classifier for each group of training data.
      iv) Sort the test data into groups according to the missing patterns.
      v) Test each group of test data using the sub-classifier trained with the training data having the same missing pattern.
      vi) Record the resulting classification accuracy, averaged across all sub-classifiers used.
5) (10-fold cross validation) Repeat the above procedure for 10 times, each time choose a different subset of test data.
6) Repeat the whole procedure for 10 times, record the average accuracy and standard deviation.
The above procedure is illustrated in Fig. 2. Note that the imputation step does not apply to the ensemble classifier treatment.

**E. Experiment Procedure: Values Missing in both Training and Test Set**

1) Randomize and stratify the data.
2) Split the data set into a training set containing the first 90% of instances, and the rest 10% forms the test set.
3) Introduce missing values into both the training set and the test set
4) Train and test the classifiers
   a) For the basic classifiers:
      i) Impute the missing values in the training data.
      ii) Train the classifiers using imputed training data.
      iii) Test the classifiers using the imputed test data.
      iv) Record the resulting classification accuracy.
   b) For the ensemble classifier treatment:
      i) Enumerate all possible missing patterns.
      ii) Sort the training data into groups according to the missing patterns.
      iii) Train a sub-classifier for each group of training data.
      iv) Sort the test data into groups according to the missing patterns.
      v) Test each group of test data using the sub-classifier trained with the training data having the same missing pattern.
      vi) Record the resulting classification accuracy, averaged across all sub-classifiers used.
5) (10-fold cross validation) Repeat the above procedure for 10 times, each time choose a different subset of test data.
6) Repeat the whole procedure for 10 times, record the average accuracy and standard deviation.

The above procedure is illustrated in Fig. 3.

Those two scenarios are experimented with to simulate practical situations i) when the recorded historical data are full, and the test data is collected through sensors and contain missing values, and ii) when the training data and test data are collected using the same type of sensors with the same failing rate.

**IV. RESULTS AND DISCUSSIONS**

In this section we discuss the findings on the performance of different missing value treatments using the above experimental platform. Our measure of performance is the average classification accuracy of 10 rounds of random introduction of missingness.

**A. Values Missing at Prediction Time Only**

![Fig. 4. Average accuracy of treatments as the percentage of missing values varies in the test set](image)

In this experiment values are missing during prediction only. Fig. 4 presents the accuracy of each treatment as the percentage of missing values changes. The accuracy reported at each missing percentage is averaged over all datasets and all classifiers. It is evident from the plot that no matter which treatment is used, the existence of missing values leads
to reduced accuracy. As the percentage of missing values increases, the performance reduction is higher. It can also be observed from the plot that the ensemble classifier treatment remains on top as the best performer.

To support the fact that the ensemble classifier treatment is the best performer in this scenario, we have obtained the number of times each treatment is the best performer in terms of absolute accuracy across different missing percentages using different classifiers and data sets. The dominance results are presented in Table IV. Each row in Table IV presents the number of times each missing value treatments are the best performers for a particular scale of missingness. There are three classifiers and nine datasets and thus the dominance values will sum to $3 \times 9 = 27$ for each row. Note that ensemble classifier treatment remains the most reliable at each scale of missingness.

We hypothesise that the ensemble classifier treatment is the best performer in this scenario, because each sub-classifier is trained with a training subset, containing all training instances with the missing attributes removed. When a test instance with missing values is presented. Imputation treatments will introduce artificial values into the test instance, this approximation error will then propagate to the prediction phase. Whereas the ensemble classifier treatment deals directly with the true observations and does not introduce this approximation error.

Fig. 5 presents the plots comparing the performance of the treatments when used with a particular classifier. These plots reveal that the accuracy trend line of the ensemble classifier treatment is always convex. This is an interesting phenomenon comparing to the decrease in accuracy of the imputation treatments, which are more or less straight lines. This phenomenon suggests that the decrease of the performance of the ensemble classifier treatment is becoming faster as the missing percentage increases, and that it suffers more severely than the imputation treatments at very large percentage of missing values.

Therefore ensemble classifier treatment works best when no values are missing in training data, and when the missingness is not too large. But which classifier yields the best performance when used as a base classifier for the ensemble classifier treatment? Fig. 6 reveals the fact that under all scales of missingness, the decision tree is the best choice. It is also clear that the performance of each classifier has an explicit ordering, which doesn’t hold for the imputation treatments, as shown in Fig. 7. The reason is that the instances passed to the base-classifiers in the ensemble classifier treatment has not been modified and all unknown values are ignored. Hence if the performance of a classifier is better than another under some missing percentage, it will always be the case. On the other hand, the imputation treatments introduce artificial values that brings along errors. Depending on the missing percentage and the imputation treatment used, the bias of the artificial data leads to varying relative performance of the classifiers.

Fig. 7 also suggests that the decision tree classifier performs best when the missingness is very small (less than 10%), but its performance decreases sharply as the missing percentage increases. It is clear that the Bayesian network classifier works the best with a large missing percentage. As the missing percentage gets large, the value of top level attributes used for branching decisions are more likely to be estimated. This might lead to completely different branches, yielding the wrong prediction. However the Bayesian network does not suffer as much because it calculates the probability of having a particular instance taking into account all attribute values, the estimation error is less likely to affect
### TABLE IV

**Superiority Summary of the Missing Value Treatments When Values Are Missing in the Test Set Only**

<table>
<thead>
<tr>
<th>Missing percentage (%)</th>
<th>Mean</th>
<th>Nearest Neighbour</th>
<th>Regression</th>
<th>Ensemble</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>4</td>
<td>0</td>
<td>8</td>
<td>15</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>22</td>
</tr>
<tr>
<td>30</td>
<td>2</td>
<td>0</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>40</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>19</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
<td>1</td>
<td>9</td>
<td>15</td>
</tr>
</tbody>
</table>

---

**Fig. 6.** Performance of various classifiers when used as base classifiers for the ensemble classifier treatment. Values are missing in test set only.

---

the final result as much.

**B. Values Missing at Both Induction and Prediction Time**

In this section we present the performance of different missing value treatments when values are missing during both the training and test phase. Fig. 8 shows the accuracy of each treatment across different missing percentage, averaged over all datasets and all classifiers. From the plot it is again evident that no matter which treatment is used, the increase in the percentage of missing values leads to greater performance reductions.

It is interesting to observe that the ensemble classifier treatment is performing the worst in this scenario. This is due to the fact that when there are missing values in the training data, the size of many training subsets will reduce. This leads to insufficient training samples for many sub-classifiers to generalise. Thus the presence of missing values in the training data reduces the overall accuracy significantly.

**Fig. 7.** Performance of various classifiers with different missing value treatments. Values are missing in test set only.
20~30%). As the missingness increases, the mean imputation performs the best. This is also confirmed by Table V, which presents the dominance summary. This can be explained by considering how the imputation treatments work. Both the nearest neighbour imputation and regression imputation rely on other available attributes in estimating the missing values. As the missing percentage increases, there are less available attributes for each instance. Thus the probability of the most relevant attributes also goes missing is higher. So the estimated value is likely to be more biased than the value estimated using mean imputation, which considers only values of the same attribute across all instances.

Looking at Fig. 9(b), we found that the nearest neighbour imputation performs more poorly than the ensemble classifier treatment. The reason is that the nearest neighbour imputation assigns the missing entries the values of its nearest neighbour. Then the nearest neighbour classifier will assign it to the same class as its nearest neighbour, which is the same instance used for imputing the missing values. Therefore this combination is just assigning the test instance the class of the instance having the most similar values for its available attributes. This is much more naive and hence has less predictive power comparing to any other treatments.

The plot in Fig. 9(c) shows the performance comparison when using the Bayesian network classifier. The plot shows that even for small missing percentages, the mean imputation performs the best. The reason being that the regression imputation estimates the missing value using the relationship between the missing attribute and other available attributes. Similarly the Bayesian network classifier considers all the attribute values of an instance and the joint probability of them occurring together. Since this classifier already takes into account the relationships between the attributes, the value estimated using regression imputation will not help much in terms of classification accuracy.

V. CONCLUSIONS

In this paper we have done an empirical study that compares the performance of various missing value treatments when (i) values are missing only during prediction phase, and (ii) values are missing during both induction and prediction phase, under different scales of missingness. Based on the findings we can reach the following conclusions:

(i) The presence of missing values always leads to performance reductions of classifier, no matter which treatment we use to deal with the missing values.
(ii) When there are no values missing in the training
Table V
Dominance summary when values are missing both in the training and test set

<table>
<thead>
<tr>
<th>Missing percentage (%)</th>
<th>Treatment</th>
<th>Mean</th>
<th>Nearest Neighbour</th>
<th>Regression</th>
<th>Ensemble</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td></td>
<td>6</td>
<td>3</td>
<td>11</td>
<td>7</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>9</td>
<td>3</td>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>13</td>
<td>2</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>40</td>
<td></td>
<td>13</td>
<td>5</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>50</td>
<td></td>
<td>17</td>
<td>3</td>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>

data, ensemble classifier treatment is the best choice. And when using the ensemble classifier treatment in such circumstances, using decision tree as the base classifier yields the best performance. Average accuracy as high as 76% can be achieved with missingness scales as large as 50%.

(iii) The introduction of missing values into the training data leads to great reduction in the performance of the ensemble classifier treatment. In such cases, the regression treatment works the best when values are missing at low percentage and the mean treatment works the best at when values are missing at high percentage.

In this particular study we assume that all the features are important for prediction. We also assume the missing values are MCAR, and the distribution of missing values is uniform across all attributes. In the future we aim to examine the performance of the treatments under the influence of other distributions of missingness over the attributes, which might be the case for many practical situations. Combination of the imputation and ignorance treatments might result in a hybrid treatment that yields consistent good performance. Further more, this study focuses on numerical attributes only. Further experiments can be conducted to see that how does the results differ if categorical attributes come into the picture.

Overall, we have demonstrated that a significant improvement in reliability can be achieved by choosing the correct missing value techniques. We also provide a guideline to facilitate the choice of missing value treatments under different scenarios.

REFERENCES