Robust Decision Tree Induction from Unreliable Data Sources

Christian Schreckenberger
Institute for Enterprise Systems
University of Mannheim
Mannheim, Germany
schreckenberger@es.uni-mannheim.de

Christian Bartelt
Institute for Enterprise Systems
University of Mannheim
Mannheim, Germany
bartelt@es.uni-mannheim.de

Heiner Stuckenschmidt
Data and Web Science Group
University of Mannheim
Mannheim, Germany
heiner@informatik.uni-mannheim.de

Abstract

The main contribution of this paper is a new criterion, called Expected Information Gain, to compute the best possible split, given that there is missing data present in a dataset. Expected Information Gain can be used to build more robust decision trees given the circumstance of missing data. We evaluate the criterion on six UCI datasets and one synthetic dataset in three scenarios: No missing data at prediction time, missing data at prediction time, and imputed data at prediction time. The results of the proposed methods are promising in all scenarios. However, especially in the second scenario the potential of learning a more robust model with the proposed method becomes apparent.

1 Introduction

The problem of missing values in training and test data is a well-studied problem in machine learning [Qui89, STP07] and different techniques for dealing with missing values have been proposed [LWTB97]. In this paper, we propose a new method for dealing with missing values that uses empirical information about the reliability of data sources. We focus on classical decision tree learning and propose a modification of the corresponding algorithm that takes the reliability of an information source into account. In particular, in contrast to the standard algorithm that selects splitting attributes solely based on the information gain achieved, we investigate combinations of information gain and source reliability as a basis for selecting attributes to split on. The underlying assumption is that sources that had a low reliability in the past will continue to produce more missing values in the future. This means that attributes from sources with a low reliability provide a less useful split and should therefore only be preferred if they provide a really high information gain to compensate for the expected missing values. The goal of the proposed method is to increase the robustness of a Decision Tree (DT).

In: Rudolph S., Marreiros G. (eds.): Proceedings of STAIRS, Santiago de Compostela, Spain, august 2020, published at http://ceur-ws.org

While the term robustness is ambiguous in our case, we refer to the definition of [RM14], where robustness is defined as the model's ability to handle data with noise or missing values.
The proposed method can be seen as an extension of the missing values strategy of the classical C4.5 DT induction algorithm. C4.5 weights the information gain with the ratio of non-missing data values for a certain attribute and assumes the examples with missing values have no contribution to the information gain. We extend this by imputing the missing values using a k-NN algorithm and compute the information gain of the imputed values and add it to the overall information gain, which now becomes a weighted sum of the gain achieved by examples with non-missing and examples with imputed data values. We can show that this improved information gain computation consistently improves the performance of DT learning on a variety of datasets.
In the following, we will introduce to the background of the proposed approach in Section 2. In Section 3 the related work to our approach is presented. Subsequently, we will define the proposed method in Section 4 and present the empirical evaluation in Section 5. Finally, we conclude this paper in Section 6.

2 Background

We base our work on the following problem definition. Let $\mathcal{D}$ be a dataset that consists of examples that are represented by the vector $(x_1, \cdots, x_n, y)$, where x_i are values provided by the attributes $A_i \in \mathcal{A}$ and $y \in \mathcal{C}$ is the class label. Further, $r : \mathcal{A} \rightarrow [0, 1]$ where $r(A_i)$ is the probability that the attribute will produce a value at any point in time.
In the following, we will explain missing data and the mechanisms that can be behind it as well as briefly introduce the C4.5 algorithm in, which we embed our proposed criterion.

2.1 Missing Data

Missing data is a common occurrence in datasets, based on the domain there may be various reasons and in some cases it is also anticipated that there will be missing data [HBB$^+$98]. [LR19] identified three mechanisms that cause the missing of data: Missing Completely at Random (MCAR), Missing at Random (MAR) and Missing Not at Random (MNAR). Based on the mechanism that causes the missing of data, different methods can be applied to impute the data that is missing. In our case, we only consider MCAR, which means that there is no relationship between the cause for the missing of data and their values.
A lot of research has been attributed to the handling of missing data. Depending on how much data is available one naive way of dealing with missing data is to discard the examples that contain missing data [End10]. However, a more sophisticated approach is to impute the values either by a simple method like mean imputation or a more sophisticated method like model based imputation. In the case of mean imputation [Zha16], every missing value of an attribute A_i is replaced by the mean of that attribute A_i. Because there is only a single value used for each attribute with missing data, this kind of method is referred to as *single* imputation. While the mean imputation is beneficial in terms of not influencing the mean of the attribute, the true variance of A_i is underestimated. An example for a model based imputation is k-NN imputation [TCS$^+$01]. Hereby, every occurrence of missing data, receives a different value, hence called *multiple* imputation. A k-NN imputation works by using the mean of the k nearest neighbors of an example. The nearest neighbors are determined by calculating the distance to other examples, based on the available values in other attributes of a given example.

2.2 Decision Tree Induction

We embed our proposed criterion in the general idea of the C4.5 Algorithm [Qui87, Qui86] for DT induction, which is shown in Algorithm 1. In the following, we will briefly explain the relevant parts of how C4.5 handles missing data during the induction and at prediction time.

2.2.1 Induction with C4.5

As shown in Algorithm 1, the induction of a DT in the C4.5 algorithm follows a divide and conquer strategy. Given the dataset $\mathcal{D}$, the algorithm iterates over every attribute $A_i \in \mathcal{A}$ and searches for the split value x_sv, where the impurity of the resulting subsets $\mathcal{D}_i \in \mathcal{D}$ is minimized according to a given criterion. This step is repeated on the resulting subsets $\mathcal{D}_i$ until a stopping criterion is met. Possible stopping criteria are that the resulting subset is pure or that there are less than two instances left in the subset.
Based on this basic DT induction algorithm, [Qui89] furthermore proposed to make adjustments to be able to handle missing data. When having missing data present during the induction of a DT two problems arise. The

Algorithm 1 `Induce_Decision_Tree`

```
Require: A labeled dataset $\mathcal{D}$, reliabilities $r$

 1: create Node $N$
 2: if stopping criterion is met for $\mathcal{D}$ then        ▷ For example: $\mathcal{D}$ is pure or $|\mathcal{D}| < 2$
 3:     store weighted class distribution in $N$, mark $N$ as leaf
 4:     return $N$
 5: end if
 6: $best\_improvement \leftarrow 0$
 7: for $A_i \in \mathcal{A}_{\mathcal{D}}$ do
 8:     for $x_i \in A_i$ do
 9:         $gain \leftarrow$ criterion($\mathcal{D}, A_i, x_i$)
10:         if $gain > best\_improvement$ then
11:             $split\_attribute, split\_value \leftarrow A_i, x_i$
12:         end if
13:     end for
14: end for
15: Store $split\_attribute, split\_value$ in $N$
16: Partition $\mathcal{D}$ into $\mathcal{D}_i$ based on $split\_attribute$ and $split\_value$
17: for each partition $\mathcal{D}_i$ of $\mathcal{D}$ do
18:     $child \leftarrow$Induce_Decision_Tree($\mathcal{D}_i$)
19:     add $child$ to $N$
20: end for
21: return $N$
```

first problem is concerned with the treatment of missing data when computing the optimal split and the second problem is concerned with the propagation of the examples that contain missing values in the selected attribute of the split. For the first problem, it was proposed to calculate the information gain on the examples with no missing data and multiply it with the reliability of the attribute to account for the missing data. In the given Algorithm 1, the criterion used in line 9 would be $r(A_i) \cdot information\_gain$. The second problem is addressed by reducing the weights for those examples where the value of the chosen attribute A_i is missing. This is done proportionally to the sum of the weights of the resulting subsets based on the chosen split.

2.2.2 Prediction with C4.5

Given a DT model and an example with no missing data the prediction is straight forward. Based on the rule in every decision node of the tree, we follow the nodes of the tree according to the values of the example until we reach a leaf, which then gives us the class distribution that reached this node during the DT induction.
However, given that the value of the example is missing for the attribute of the rule in a decision node, the C4.5 strategy to handle this issue is to propagate the example to all successive decision nodes and then weight the class distributions proportionally to the weights of the splits.

3 Related Work

In this section we present the related work of our approach. However, as we already mentioned the C4.5 algorithm in Section 2, we will not mention it here again.
In [Fri76], the first idea of how to deal with missing data in DTs was introduced. During the induction of the DT, they propose to ignore the samples with missing values for an attribute and calculate the impurity reduction only based on those examples that have values present for a given attribute. Once the split is chosen, the samples with missing values in the chosen attribute are then passed down to both child nodes. Another rather trivial method was used in [CN89]. In this paper, the most common attribute value was used for imputation in an attribute. Following this, they apply their proposed learning algorithm to induce the DT.
Another, DT induction algorithm that can handle missing data is the so called lazy decision tree [FKY96]. Hereby, the DT is induced on-the-fly, based on the attributes that have no missing values at prediction time. The apparent downside of this approach is that it is slow in the inference because the DT is learned just in time with the arrival of the example.

In [BFOS84], they propose to use a so called surrogate split. Hereby a primary split is calculated for a given node, based on this primary split, a list of surrogate splits is calculated and ranked based on the fact of how closely they resemble the distribution of the primary split. If during prediction time, the value for the primary split is missing, the algorithm falls back to the list of surrogate splits. According to [Ste09], the CART algorithm later adapted the mechanisms of C4.5 to handle missing data.

The way of dealing with missing data of [Loh09] is, that all non-categorical values are sent to the left node, but if enough missing data is present in the data the algorithm may use the missing of the value as a value to split upon. If data is missing during prediction time, the data is classified according to the majority rule.

An approach, designed to handle MAR data in the DT induction is introduced in [BR18]. In this paper, the authors present a method called BEST (Branch-Exclusive-Splits Trees). Hereby, only attributes are selected that have no missing data at the current node. However, as the partitioning of the data progresses, regions with no missing data will appear that could not be considered beforehand.

In [TJH08] an approach called MIA is introduced. The authors state that the MIA can be plugged into any DT induction algorithm regardless of splitting, stopping or pruning rules. Based on MIA, there are three possible ways to split when missing data is present. Either the examples with missing data are included on the left side of the split, the examples with missing data are included on the right side of the split, or a split is made on if data is missing or not.

4 Expected Information Gain

Given a DT induction algorithm such as C4.5, we propose a novel criteria that can be used to determine the optimal split for a DT, in case missing data is present. The name of the proposed criterion is called *Expected Information Gain* (EIG). In the following we will define EIG and the formulas on which EIG is grounded.

Definition 1 (Entropy) *Let $\varphi(y)$ be the weight function, then the Weighted Entropy of a dataset $\mathcal{D}$ is given by*

$$H(\mathcal{D}) = -\sum_{y\in\mathcal{C}} \varphi(y)P(y)\cdot log_2 P(y)$$

Let $\mathcal{D}$ be a dataset and $D_{(A_i,x_i)} \subseteq \mathcal{D}$ the subset of $\mathcal{D}$ with $A_i = x_i$. A weight function is of $\mathcal{D}$ and $D_{(A_i,x_i)}$ is given by φ. The conditional Entropy of a dataset given information from Attribute A_i is defined for categorical attributes as

$$H(\mathcal{D}|A_i) = \sum_{x_i\in A_i} \frac{\varphi(D_{(A_i,x_i)})}{\varphi(\mathcal{D})}\cdot H(D_{(A_i,x_i)})$$

and given a split value x_{sv} for continuous attributes

$$H(\mathcal{D}|A_i) = \frac{\varphi(D_{(A_i,x_i\leq x_{sv})})}{\varphi(\mathcal{D})}\cdot H(D_{(A_i,x_i\leq x_{sv})}) + \frac{\varphi(D_{(A_i,x_i> x_{sv})})}{\varphi(\mathcal{D})}\cdot H(D_{(A_i,x_i> x_{sv})})$$

In our approach we rely on the weighted entropy to measure the impurity of a given subset of the data. The need for using the weighted entropy, instead of the normal entropy, arises from the propagation of the examples with missing values. Due to this propagation the weights are not equal for all examples and therefore need to be taken into account.

Definition 2 (Information Gain) *Let $H(\mathcal{D})$ and $H(\mathcal{D}|A_i)$ be the Entropy and Conditional Entropy as defined above. The Information Gain associated with a Attribute A_i is defined as*

$$IG(\mathcal{D},A_i) = H(\mathcal{D}) - H(\mathcal{D}|A_i)$$

For the information gain we follow the well known definition, where the conditional entropy of a possible split is subtracted from the entropy that is present in the current subset of the data. Based on these definitions, we can now define the Expected Information Gain.

Definition 3 (Expected Information Gain) *Let $IG(\mathcal{D},A_i)$ be the information gain associated with attribute A_i as defined above and $r(A_i)$ the reliability of A_i, then the Expected Information Gain of A_i is defined as*

$$EIG(\mathcal{D},A_i) = r(A_i)\cdot IG(\mathcal{D},A_i^{not\_missing}) + (1-r(A_i))\cdot IG(\mathcal{D},A_i^{imputed})$$

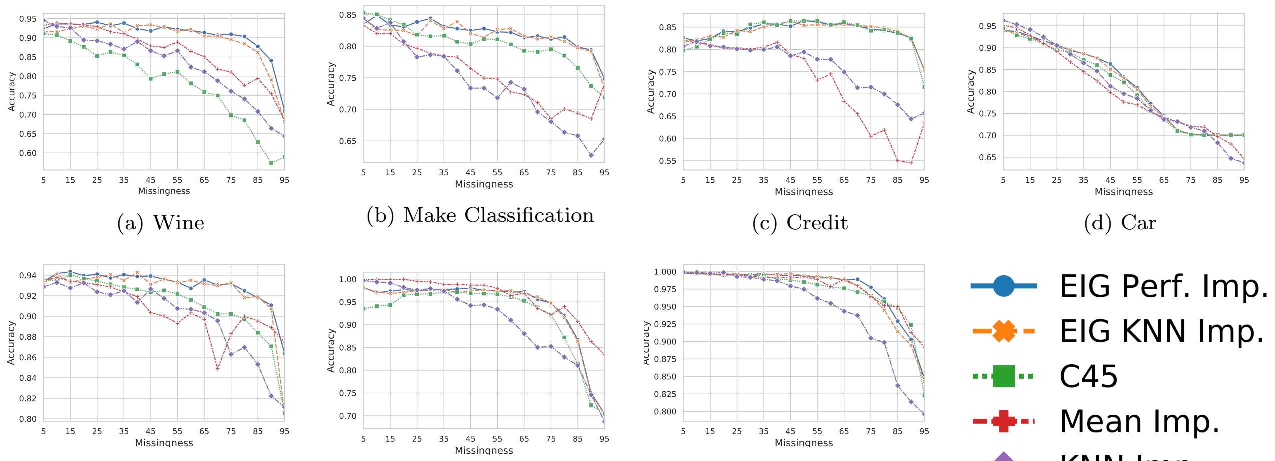
The EIG is made up of two terms, the first term is the information gain applied to the subset of the dataset at the current decision node where no data is missing. In the second term of the formula, the information gain function is applied to the subset of the dataset where data was missing originally for attribute A_i, but was imputed by a given multiple imputation method, such as k-NN imputation. Given the DT induction algorithm in Algorithm 1, the EIG would be used in line 9 and replaces the initial idea of C4.5 of multiplying the Information Gain of the dataset with the reliability of the attribute. The reliability of an attribute $r(A_i)$, is measured as the percentage of missing data in that attribute A_i.

5 Evaluation

We use six datasets from the UCI Machine Learning Repository and the synthetic data generation `make_classification` from scikit learn with standard parameters to evaluate the performance of the proposed EIG. In our implementation we use a binary decision tree[1]. Therefore, all non-binary categorical features in the datasets are one-hot encoded. In the evaluation, the learning is always performed on datasets with missing data, i.e. the training data has missing data. However, for the test set we accounted for three different scenarios, i.e. test set has no missing data, test set has missing data, and the missing data in the test set is imputed. In the third case, a k-NN imputer is fitted on the training set and then used on the test set to fill the missing data. The missing data is introduced randomly to the datasets, meaning that every feature has about the same probability of having missing data as the others. We hereby evaluate in a range from 5% to 95%, using steps of 5%. For every percentage of missing data, the missing data is randomly generated five times and for each run we use a 5-fold stratified cross validation.

In the experiments, we consider two variants of the proposed EIG and three baselines. The two EIG variants make use of the k-NN imputation and what we call perfect imputation. In the latter imputation method, we use the actual data values and consider them to be imputed. The three baselines are all based on the C4.5 algorithm, the first baseline uses the proposed C4.5 way of handling missing data during the training process, the second and third baseline use mean and k-NN imputation respectively on the training data and then learn a DT as if no data is missing. In all cases of k-NN imputation: $k = 5$.

5.1 Prediction with Full Data



(a) Wine (b) Make Classification (c) Credit (d) Car

(e) Breast Cancer (f) Autism (g) Audit

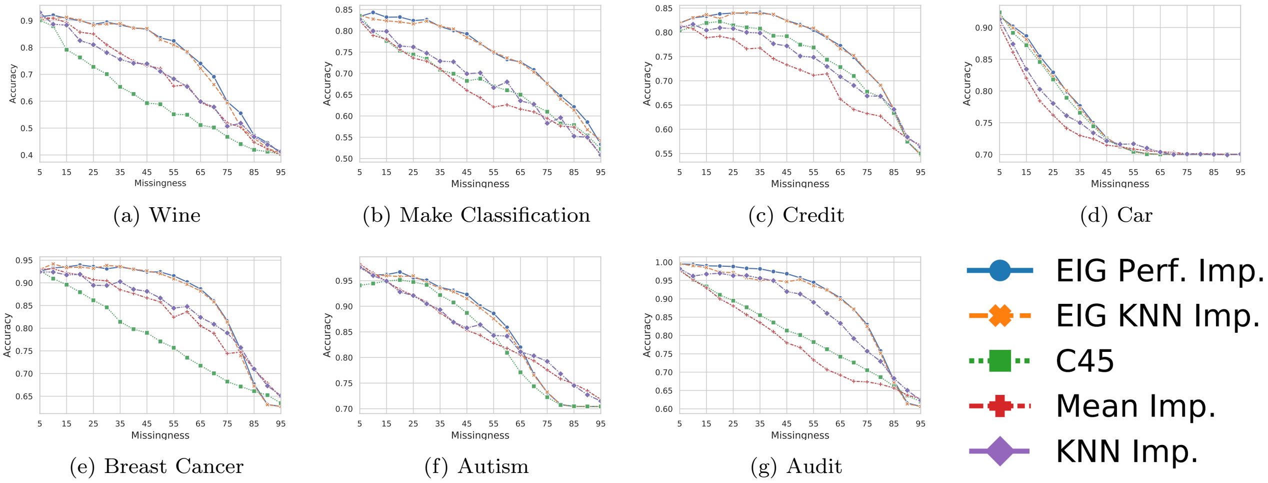
Figure 1: The results for the seven datasets, when the test set has no missing data

The results for the first experiment setting, i.e. predicting with a test set that has no missing data, are shown in Figure 1. The missingness values in this case describe the amount of missing values in the training data. Considering the mean over all tested missingness values, the two EIG variants perform better than the baselines in five of the seven datasets. In these five cases the difference between the best baseline and the EIG variants are larger compared to the Audit and Autism datasets, in which the k-NN baseline outperforms the two EIG

[1] The code is available at github.com/cschreck/dt-eig

variants only by a small margin. From Figure 1 it can be observed that the accuracy curves for the EIG variants are quite similar. In a few instances the k-NN imputation performs better than the perfect imputation, however, considering the mean of all values, it shows that the perfect imputation has a small advantage over the k-NN imputation when used with the EIG.
In the Wine dataset the two EIG variants start to perform significantly better at around 35% of missingness in the data. This is mainly because the baselines keep on losing accuracy, while the two EIG variants perform steady until 90% of missingness. A similar pattern can be observed in Figure 1b and 1e. An interesting occurrence is that the accuracy for the two EIG variants as well as the C4.5 baseline increase with more missing data in the Credit dataset. The two imputation based baselines perform better on the Autism and Audit datasets, which seem to be very learnable as they almost reach a perfect accuracy for low missingness values.

5.2 Prediction with Missing Data



(a) Wine

(b) Make Classification

(c) Credit

(d) Car

(e) Breast Cancer

(f) Autism

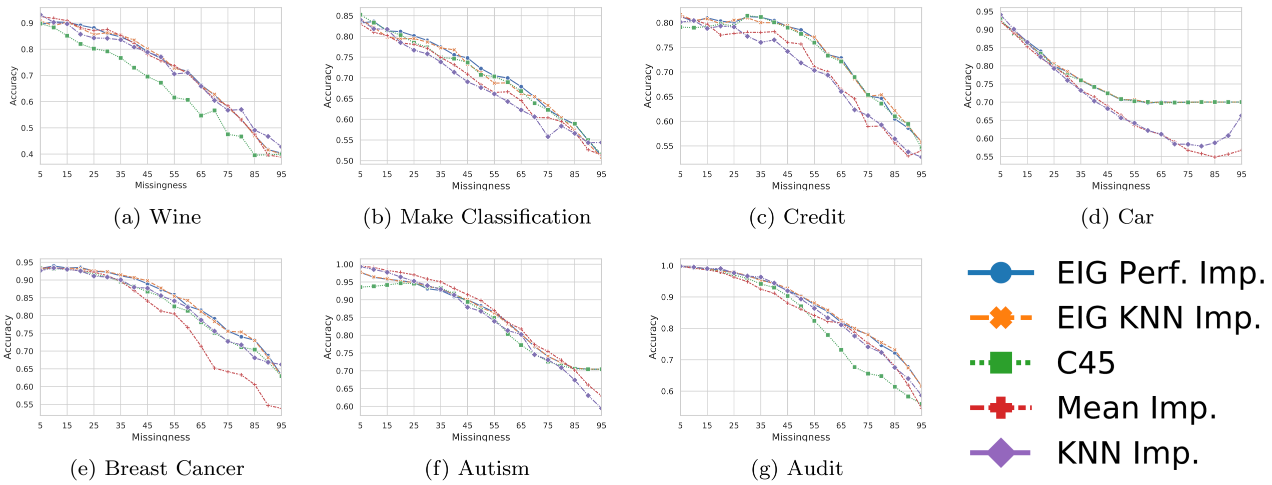
(g) Audit

Figure 2: The results for the seven datasets, when the test set has the same missingness as the training set

Figure 2 depicts the results for the second scenario where not only during training time the data is missing but also during prediction time. In this scenario, the two EIG variants perform better than the baselines in terms of the mean accuracy per dataset on all datasets. For the two EIG variants, the perfect imputation performs better than the k-NN imputation on all datasets when considering the mean accuracy. However, these two variants are quite close in their performance and on a few occasions using the k-NN imputation in the EIG, actually provides better results than the perfect imputation.
While the mean accuracy of the five methods are close on the two datasets Car and Autism, a significant increase of mean accuracy can be observed in the Wine, Credit, Breast Cancer, Audit and synthetic Make Classification dataset. Compared to the C4.5 baseline, the two EIG variants only perform worse with high missingness values. In the Car and Autism dataset, the accuracy of the two EIG variants is approaching the accuracy of the C4.5 baseline until they are more or less equal. Another observation that can be made in the Autism dataset is that the two EIG variants perform better initially but with more missingness introduced to the dataset they fall below the imputation baselines at 70% of missingness.

5.3 Prediction with Imputed Data

In Figure 3, the results for the third scenario are shown, where the missing values of the test set were imputed. Judging by the mean accuracy over all missingness values, the two EIG variants perform better in six out of the seven datasets. The performance is worse than the mean imputation method on the Autism dataset. Analyzing the performance of the two EIG variants, it can be observed that there is barely a differences between them. Using the k-NN imputation for the EIG in this scenario is better in four out of the seven datasets.
The curves are mostly similar for all methods on the used datasets. For the Wine dataset it can be observed that the C4.5 approach performs significantly worse than the four other methods, the mean accuracy of it is about seven percent points lower compared to the other methods. On the Car dataset it can be observed that at a missingness of 55% the accuracy of the two EIG variants and the C4.5 algorithm become a flat line of 70%



(a) Wine (b) Make Classification (c) Credit (d) Car

(e) Breast Cancer (f) Autism (g) Audit

Figure 3: The results for the seven datasets, when the missing data of the test set is imputed

of accuracy even with the missingness further increasing. The 70% in this dataset represents the majority class, meaning that in this case the C4.5 and the two EIG variants resort to a prediction of the majority class, while the other two baselines seem to be impacted negatively.

5.4 Discussion

Comparing the three scenarios amongst each other it can be observed that the proposed approach performs best compared to the baselines in the scenario where we also have missing values present during prediction time. This means that the learned model is more robust when using the EIG in terms of its ability to handle missing data at prediction time. But still, even when there is no data missing at prediction time or the data is imputed, using the EIG proves to be beneficial in most cases. As the perfect imputation method outperforms the k-NN imputation for the EIG in a majority of the cases, we assume that a more accurate imputation method generally provides better results. Analyzing the trees that are built with the C4.5 approach and the two EIG variants, it shows that especially the propagation of the examples with missing values lead to deeper trees compared to the trees in the two imputation based baselines.

However, it has to be noted that since the proposed EIG needs an imputation method such as k-NN imputation, that relies on dependencies among the features to impute the missing data, there would be no benefit in applying EIG to a dataset with features that are independent from one another. This means that attention has to be paid to the underlying structure of the data when applying the proposed DT induction with EIG.

6 Conclusion

In this paper, we introduced the well-known problem of missing data in machine learning. In the following, we described the background of the problem, especially with regard to the C4.5 algorithm. Subsequently, we introduced the Expected Information Gain, which substitutes the approach of C4.5 to handle missing data. The evaluation contained three scenarios where the performance of the EIG was tested in case, there was no missing data, missing data, or imputed data at prediction time. The results have shown to be beneficial for our proposed method. We could especially demonstrate an increasing robustness in the induced DT when there was data missing at prediction time.

Future work should comprise a thorough analysis on the impact of a given imputation method on the result. This analysis should also contain the application of various other imputation methods that are applicable to the proposed approach. Furthermore, stopping criteria as well as pruning methods should be investigated especially with regards to the examples that are propagated with missing values.

Acknowledgments

This research was supported by the German Federal Ministry for Economic Affairs and Energy (Grant No. 01MD18011D).

References

[BFOS84] L Breiman, JH Friedman, R Olshen, and CJ Stone. Classification and regression trees. 1984.

[BR18] Cédric Beaulac and Jeffrey S Rosenthal. Best: A decision tree algorithm that handles missing values. *arXiv preprint arXiv:1804.10168*, 2018.

[CN89] Peter Clark and Tim Niblett. The CN2 induction algorithm. *Machine learning*, 3(4):261–283, 1989.

[End10] Craig K Enders. *Applied missing data analysis*. Guilford press, 2010.

[FKY96] Jerome H Friedman, Ron Kohavi, and Yeogirl Yun. Lazy decision trees. In *AAAI/IAAI, Vol. 1*, pages 717–724, 1996.

[Fri76] Jerome H Friedman. A recursive partitioning decision rule for nonparametric classification. *IEEE Trans. Comput.*, 26(SLAC-PUB-1573-REV):404, 1976.

[HBB$^+$98] Joseph F Hair, William C Black, Barry J Babin, Rolph E Anderson, Ronald L Tatham, et al. *Multivariate data analysis*, volume 5. Prentice hall Upper Saddle River, NJ, 1998.

[Loh09] Wei-Yin Loh. Improving the precision of classification trees. *The Annals of Applied Statistics*, pages 1710–1737, 2009.

[LR19] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.

[LWTB97] Wei Zhong Liu, Allan P White, Simon G Thompson, and Max A Bramer. Techniques for dealing with missing values in classification. In *International Symposium on Intelligent Data Analysis*, pages 527–536. Springer, 1997.

[Qui86] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[Qui87] J Ross Quinlan. Decision trees as probabilistic classifiers. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 31–37. Elsevier, 1987.

[Qui89] J Ross Quinlan. Unknown attribute values in induction. In *Proceedings of the sixth international workshop on Machine learning*, pages 164–168. Elsevier, 1989.

[RM14] Lior Rokach and Oded Maimon. *Data Mining with Decision Trees: Theory and Applications*, volume 81 of *Series in Machine Perception and Artificial Intelligence*. WORLD SCIENTIFIC, 2 edition, October 2014.

[Ste09] Dan Steinberg. Cart: classification and regression trees. In *The top ten algorithms in data mining*, pages 193–216. Chapman and Hall/CRC, 2009.

[STP07] Maytal Saar-Tsechansky and Foster Provost. Handling missing values when applying classification models. *Journal of machine learning research*, 8(Jul):1623–1657, 2007.

[TCS$^+$01] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

[TJH08] BETH Twala, MC Jones, and David J Hand. Good methods for coping with missing data in decision trees. *Pattern Recognition Letters*, 29(7):950–956, 2008.

[Zha16] Zhongheng Zhang. Missing data imputation: focusing on single imputation. *Annals of translational medicine*, 4(1), 2016.