Impact of Ensemble Machine Learning Methods on Handling Missing Data

Ernest Perkowski
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
e.perkowski@student.utwente.nl

ABSTRACT

Missing values are a common problem present in data from various sources. When building machine learning classifiers, incomplete data creates a risk of drawing invalid conclusions and producing biased models. This can have a tremendous impact on many business sectors or even human lives. Ensemble methods are meta-algorithms that can combine weak base estimators into stronger classifiers. Ensemble learning can make use of both ML and non-ML techniques. Using this approach proved to yield better predictions in many use cases. This research examines various usages of ensemble methods for handling missing data. Moreover, the impact of using ensemble learning is explored, given various levels of test data artificially generated based on missing at random (MAR) mechanism.

Keywords

Data Cleaning, Data Cleansing, Missing Data, Machine learning, ML, Ensemble, Bagging, Boosting, AdaBoost

1. INTRODUCTION

Data cleaning is a tedious and time-consuming process that aims for discovery and removal of erroneous, incomplete, inconsistent, and many other types of noise in order to improve the quality of the data [9]. It is believed that this step of data processing is takes most of the time needed for data analysis [15]. In order to use predictive models to search for insights, the data should be complete. This is often not the case, as missing values are a common problem introducing bias that impacts the models trained on them. Biased data leads to biased models. The seriousness of this problem depends partly on how much data is missing, the pattern of data missingness and its underlying mechanism. There are three main ways to cope with incomplete data. The first and the least effective [19] is by removing the rows with null values. The second includes various imputation techniques such as ad-hoc mean or median substitution, which are considered traditional. More advanced solutions from this category are multiple imputations, maximum likelihood or expectation maximization [1]. The third one focuses on predictive machine learning models, which tend to yield good results [2].

33^{th} Twente Student Conference on IT July. 3^{rd}, 2020, Enschede, The Netherlands.

Due to the popularity of the problem, there is an extensive research on the various approaches to handle missing values. The main focus of this paper is to examine different ensemble learning techniques, their application, and performance impact on handling missing data. In particular, the following questions will be explored:

RQ1 What is the state of the art of ensemble methods used for handling missing data?

RQ2 What is the impact of using ensemble machine learning methods, in terms of model fit, on various test data sample sizes?

To answer the above mentioned questions, a literature review is conducted and some of the ensemble methods used by other researchers will be described. Then, a number of experiments is conducted on two separate datasets. The missing values will be introduced using a generative process described further in this paper. Some of the most common ML algorithms for solving regression and classification problems are trained and used to predict previously generated missing values. The percentage of data missingness ranges from 1-100% relatively to test data size.

This paper is divided into the following sections. In the Background section, an explanation of key concepts and methods from ensemble learning and missing data mechanisms is given. Related Work describes the discoveries made by researchers working on missing values imputation together with ensemble. This is followed by a discussion on Methodology and Results of conducted experiments aiming to discover the impact of using ML ensemble models on various levels of missing data.

2. BACKGROUND

2.1 Ensemble methods

The core idea of ensemble decision making is present in our daily lives. We seek others' ideas about a problem and then evaluate a few different opinions in order to draw the most optimal conclusions. Ensemble learning aim to improve ML performance by combining a collection of weak classifiers into a single stronger classifier [4], [22]. Thereafter, a new instance is classified by voting the decision or averaging in regression. Below, an explanation of certain ensemble methods used later in the experiments, is given:

2.1.1 Bagging

Bagging, also called bootstrap aggregating, was introduced in 1996 by Breiman [3]. This method is used for improving unstable estimations or classification problems. Bagging is a technique of variance reduction for given base learners, such as decision trees, or variable selection methods used for linear model fitting. Bagging generates additional data for training from the original dataset, using combinations with repetitions to create multisets with the same data

structure as the original set.

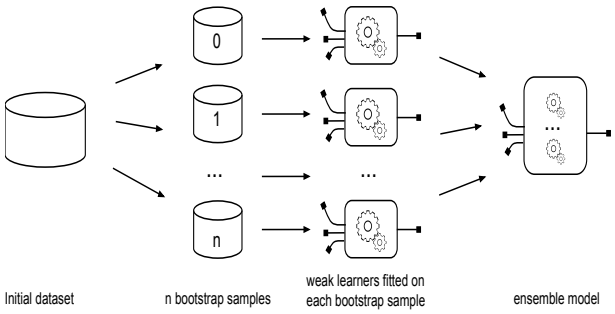


Figure 1: *Graphical representation of Bagging.*

2.1.2 *Boosting (AdaBoost)*

Boosting is a similar approach to Bagging. The core idea is to build a family of models that later on will be aggregated and compose a stronger learner, capable of better performance. The main difference between Bagging and Boosting is the sequence of performing the tasks. In Bagging, fitting the models is done in parallel and independently, while in Boosting it is done sequentially and each next model depends on the models fitted in previous steps. At every step, more focus is directed at the observations that were poorly handled by the previous model, which results in a strong classifier with lower bias. AdaBoost is a modified Boosting algorithm, it keeps track of, and updates the weights attached to each of the training set observations. The weight determines the observations to focus on.

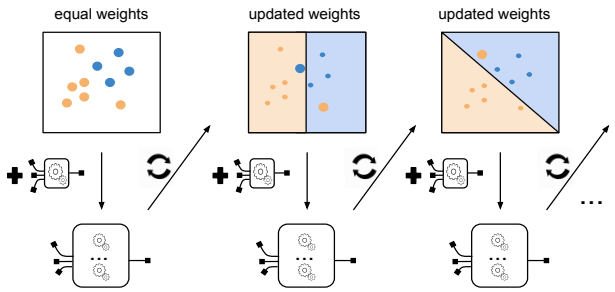


Figure 2: *Graphical representation of AdaBoost.*

2.2 Missing Data Mechanisms

When thinking about data, it is important to make a distinctions between different types of the missing data randomness. They are crucial to keep in mind, as they determine which statistical treatments of the missing data can be effectively applied. We can distinguish between three main mechanisms [16]:

2.2.1 *MAR*

Data *missing at random* (MAR) refers to a collection, where instances with and without missing values have a systematic relationship [7]. This can be simply explained with an example from medical data. If there is an emergency, there is a tendency that some details are omitted when filling in a medical form, compared to a situation of scheduled appointment with the doctor. In the former situation, the time is critical and the patient might not be able to provide all the required details which yield a relationship.

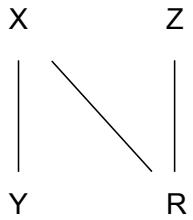


Figure 3: *Graphical representation of MAR [17]. X represents variables completely observed, Y partly missing, Z represents component that causes missingness unrelated to X and Y, R represents the missingness.*

2.2.2 *MCAR*

Data *missing completely at random* (MCAR) represents the variables that are completely unrelated either to values of the specific variable, or other measured variables. Compared to MAR, it is more restrictive as there is no correlation between missing data. Such a mechanism often occurs in real-world situations [7]. For example, students can obtain MCAR exam results due to unforeseen circumstances that cause the mechanism e.g. family situation, funeral, illness.

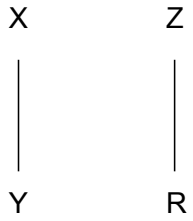


Figure 4: *MCAR [17].*

2.2.3 *MNAR*

When the data missingness is neither MAR nor MCAR but still systematical, it is referred to as data *missing not at random* (MNAR). In this mechanism, there is a relationship between the missing variable and its values [1]. Suppose there are students that experience test anxiety and have missing test scores due to the fact that they could not carry on with the exam.

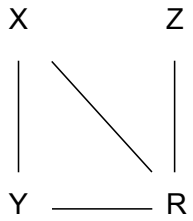


Figure 5: *MNAR [17].*

3. RELATED WORK

Missing data handling techniques have been studied extensively in the literature. The most well known include various types of imputations (e.g., [6], [8], [5]). This review will focus primarily on ensemble learning approaches to handling missing data.

As one of first studies in this field, Optiz D. et al., performed an extensive research on over 20 datasets, using both neural networks and decision trees as classifiers for

ensemble methods. As a result, it was found that in majority cases Bagging offers more accurate predictions than an individual classifier, while in some, it yields much less accurate than Boosting [12]. This, amongst others, gave the ground for future research, by showing the capabilities of such techniques.

Twala et al. proposed an ensemble of Bayesian Multiple Imputation (BAMI) and Nearest Neighbour Single Imputation (NNSI). The separate results of both algorithms are fed to decision trees and further evaluated. It has been discovered that such combination improved the accuracy compared to the baseline imputation method (BAMI and NNSI) [19]. Shortly after, another study on ensemble followed. This time the objective for was to compare 7 various missing data handling techniques (MDT) as well as an ensemble learning of two MDTs. All of the techniques used in the study are non-ML and the results show that an emsemble of expectation maximization multiple imputation (EMMI) together with C4.5 [14] yields superior performance compared to individual MDTs [21]. Later, Twala and Cartwright proposed a novel approach based on bootstrap sampling, where incomplete data is split into subsamples and fed into a decision tree classifier. The resulting ensemble consists only of decorrelated decision trees and uses them as input to make a decision [20]. The authors concluded by explaining that the proposed strategy potentially can improve prediction accuracy, especially if used in combination with multiple imputation.

Lu et al. conducted a study, where the use of Bagging and Boosting is used for continuous data imputation purposes. The study compares KNN and logistic regression to the earlier mentioned ensemble methods and finds that the more sophisticated approach underestimates variance compared to true data, but in a significantly lower degree than the individual regressors [10].

A different approach, based on a random subspace for multiple imputation method, was proposed by Nanni et al. Their idea is to put the missing values into different clusters of random data and calculate their value using the mean of the cluster or the center. This technique requires several iterations on the random subspace to create an ensemble. The authors compare several ensemble and classifier systems on various medical datasets and show that the proposed approach outperforms other existing techniques of missing data handling on numerous datasets and the performance does not drop on data missingness up to 30% [11]. Tran et al. used a combination of multiple imputation and ensemble learning to build a diverse ensemble of classifiers which then was used for predicting the incomplete data. The study focused on random forest as a regression method and compared the accuracy to other single imputation methods such as hot deck and KNN-based. From the results it is clear that the ensemble of multivariate imputation by chained equations utilising the earlier mentioned regression methods yields the best accuracy [18].

As outlined in the literature review, typical solutions to missing data problem include various imputation methods algorithms, which estimate the missing variable based on other observed values of that variable. Due to the sensitivity of individual imputation techniques to significant errors in estimation, especially for large dimensional datasets, ensemble methods have been employed.

4. METHODOLOGY

The objective of the experiments created in this research is to discover the significance of ensemble model fit gain when evaluated on missing data prediction compared to individual ML algorithm. The effects of different proportions of missing data when classifying new instances are further evaluated. This section describes the complete project setup and steps required to successfully generate missing values, train models, test and measure the performance of ML algorithms as well as ensemble learning.

As outlined in the Background section, missing data mechanism is an important aspect of data imputation. For this study, MAR has been selected as the relevant type of data missingness due to its wide occurrence in real-life datasets. The datasets chosen for this research are both small and large, and contain a mix of numerical and categorical variables. The data does not have any missing values, as it was crucial to have a total control over the whole datasets.

The experiments were conducted in Python programming language, using PyCharm environment [13]. The data was processed and handled using Pandas library and the graphs were visualized using Matplotlib library. To apply the machine learning models on data, sci-kit learn was used.

4.1 Data

To carry out the experiment, the following datasets were used:

- Avocado Prices (retrieved from https://www.kaggle.com/neuromusic/avocado-prices)
- Hearth Disease (retrieved from https://www.kaggle.com/ronitf/heart-disease-uci)

These datasets were chosen to provide different perspectives on the results obtained from the experiments. Avocado Prices contains longitudinal data on avocado sales. The dataset is quite large, as it consist of around 19000 rows. On the other hand, Heart Disease dataset has only 303 rows and the data comes from the healthcare domain.

4.1.1 Data Preprocessing

To ensure that the results are correct, the data was scaled before any computations, data splitting or model fitting. This is an essential step for ML algorithms that base their predictions on distances between data points. To avoid any features dominating over others when calculating the distances, the data needs to be scaled as some features have a higher value range than others. This was done using sci-kit learn StandardScaler function, which essentially transforms the features, so that their distributions have a mean value 0 and standard deviation of 1. The standarization function could be defined as follows:

$$z^n = \frac{x - \mu}{\sigma}$$

4.1.2 Generating MAR Data

To evaluate the performance of algorithms on predicting the MAR data, a process of introducing empty values to a complete dataset has been created.

First, a target attribute has been selected and split from the rest of the data. To simulate that the missing value is only dependent on the data observed, the weight (W) matrix has been defined. Its dimensions are based on the dimensions of target attribute matrix. The matrix W has been filled with artificially created float-type variable. This variable could not be correlated with any other variable, other than target attribute, present in the dataset in order to meet the MAR mechanism requirements. The matrix W has been filled by values randomly drawn from a uniform distribution over [0,1) and assigned a positive correlation to the probability of missingness of target attribute. These steps assure that MNAR mechanism would

not be achieved by mistake, as the target attribute is not given a correlation to any other variable in the dataset.

Having the W matrix and probability of missingness, we can successfully introduce missing values to the target attribute by comparing the randomly generated weight with the conditional missing probability. This process is repeated several times in order to allow for generation of high percentages of missing data. The amount of NaN values is determined by a threshold value which is assigned before the algorithm run.

4.1.3 *Train/Test Split*

To maintain a stable amount of data for the training set, while changing the amount of missing values in each iteration of model training and testing, two subsets of data were created. From the entire collection, half of the rows were sampled to be used for the training set. The remainder has served as a base set for generating missing data. In each iteration, a new percentage of missing values have been introduced and the rows containing null values were selected and used as testing set, to evaluate score of the algorithms.

4.2 Algorithms

Several ML models have been chosen to conduct the experiments. They have been divided into two categories, where the usage depends on the variable type. These models were selected because they are the most widely used for regression and classification problems.

Table 1: *Models selected*

Numerical Variables	Categorical Variables
Linear Regression	Logistic Regression
Bayesian Ridge Regression	Perceptron
Decision Tree Regressor	Decision Tree Classifier
K-Nearest Neighbors Regressor	K-Nearest Neighbors Classifier

4.2.1 *Model Evaluation*

The setup of the experiments needed to tackle regression and classification problem. For this reason, in each dataset, one categorical and one numerical attribute was selected. To perform the predictive modelling, depending on the type of variable, an appropriate algorithm was used. The ensemble methods and individual ML models performance were assessed using precision and recall for categorical variables. On the other hand, numerical values were assessed using scikit-learn r^2 score function.

4.2.2 *Hyperparameter Tuning*

Decision Tree and KNN algorithm performance is highly impacted by the parameters used for fitting the models. To ensure that the models are trained with optimal parameters, GridSearchCV from sci-kit learn has been used for the evaluation. GridSearchCV takes an array of possible parameters and tests the performance of model with each combination of parameters. Based on the scores, it returns the most optimal combination. Hyperparameters used in the GridSearchCV:

Table 2: *Decision Tree Hyperparameters*

Parameter	Values
criterion	gini, entropy
splitter	best, random
max_depth	2, 3 .. (training samples) -1
min_samples_split	2, 3 .. 12

Table 3: *KNN Hyperparameters*

Parameter	Values
n_neighbors	2, 3, .. 12
weight	uniform, distance
algorithm	auto, ball_tree, kd_tree, brute
leaf_size	12, 20, .. 100
p	1, 2 .. 10

4.2.3 *Ensemble Methods*

Bagging and AdaBoost are the two ensemble methods used to conduct this study. The functions were applied from scikit learn library and used with default parameters. Since one of the ML algorithms evaluated is Decision Tree, Random Forest has been added to this study as well. RF is a bagging method, which creates an ensemble of decision trees with large depths. Moreover, the algorithm makes use of random feature selection subspace for more robust models. The implementation of AdaBoost for KNNClassifier was not possible using sci-kit.

5. RESULTS

The graphs visualize model fit performance score for different ML models and ensembles. The plots express score obtained by a specific algorithm either using R^2 (for numerical variable), or f1 score (for categorical variable). Each of the graphs contain a legend explaining which color symbolizes a specific algorithm. Some of the visualizations showing the most significant impact of ensemble methods can be seen below, while remaining graphs can be found in the Appendix section.

From the selected algorithms, all performed well (more than 90% R^2 score on average) in the classification problem on Avocado dataset. Furthermore, in the Avocado dataset, we can see high performance (around 80% R^2 score) of Decision Tree Regressor, with the ensembles yielding improvement of over 10-15% compared to the base learner (see **Figure 6**). KNN Regression scores similarly to Decision Tree and its AdaBoost ensemble, while Bagging slightly improved the results, giving on average 2-3% increase (see **Figure 7**). KNN Regression scores. Both Linear Regression and Bayesian Ridge score very similarly to each other (around 58% on avg., see Appendix), with Bagging giving almost the exact same results as the base learner, and AdaBoost scoring significantly lower than the base learner.

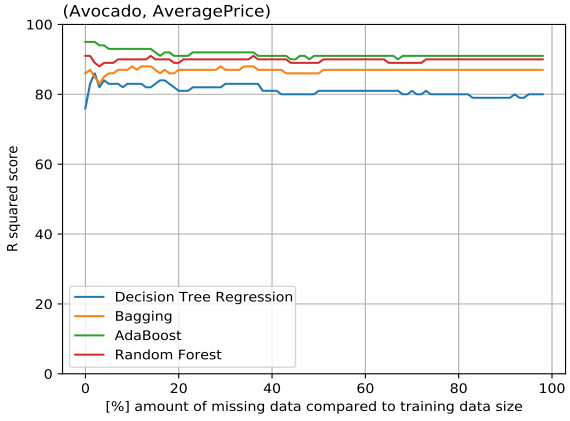


Figure 6: *Decision Tree Regressor and its ensembles on 'Average Price' attribute from Avocado dataset*

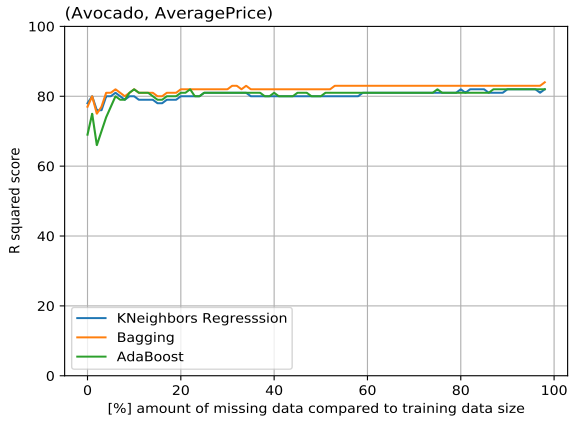


Figure 7: *KNNeighbors Regressor and its ensembles on 'Average Price' attribute from Avocado dataset*

Contrary, the results of models fit on Heart Disease dataset are significantly lower. For the regression problem, the most readable and interesting results we can notice once again for the Decision Tree. As one can see, some values of R^2 for the base learner are negative, while the ensemble methods score much better (see **Figure 8.**). In classification problem, Decision Tree scores on average similarly to its ensemble methods (see **Figure 9.**).

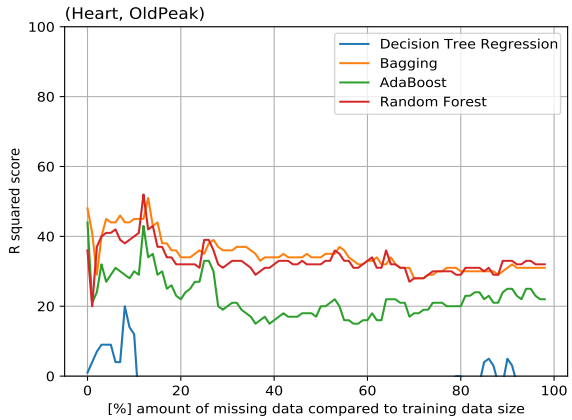


Figure 8: *Decision Tree Regressor and its ensembles on 'Old Peak' attribute from Heart Disease dataset*

Figure 9: *Decision Tree Classifier and its ensembles on 'Restecg' attribute from Heart Disease dataset*

6. DISCUSSION

To begin with, as we can see in the results from the experiments, the regression models performed much better overall on the Avocado dataset, than on the Heart Disease. The model fit score is significantly higher and the results are more stable across various amounts of missing data. This is mostly due to the size of the collection. Since it is large enough, the algorithms can gain valuable insights and fit the model on data that represents the largest amount of collection. The classification models obtained high scores on the Avocado dataset as well.

Bagging offers very little to no model fit score improvement for Linear Regression and Bayesian Ridge Regression on both datasets. When used for KNNeighbors Regression, it resulted in a few percent increment over the base learner.

The behavior of Adaboost is rather quite unexpected, as the use of this method should yield prediction accuracy at least equal to the base learner, or higher. This might be due to a possible error in the implementation.

Perceptron and Logistic Regression scored over 90%, with their Bagging yielding very similar results to the base learner. Adaboost on average gave very similar score to the base learner for Perceptron, while it performed slightly worse for Logistic Regression. Decision Tree Classifier and all the ensembles resulted in F1 score of 1, giving the best possible performance on various amounts of testing data. KNN performed similarly, having just small deviations from the maximum F1 score.

The results of the experiments on the Heart Disease dataset are unstable, as earlier mentioned, due to the size of the dataset. Perhaps the data points drawn to the training set are not well representative of the whole collection. Since the models are poorly fit, given the same training set for every testing set, the score results differ significantly for various levels of data missingness. In the case of KNN Regression and Decision Tree Regression, the model fit performance significantly drops given more missing values to predict, which was expected.

The unstable results can be explained by experimental setup. Each testing set is created by the generative procedure outlined in Generating MAR Data section. Every iteration generates missing values in most likely different rows of data, creating different training data samples. Based on the training set, different testing data samples might result in significantly different performance of the model. This might be due to the model being fitted better to certain data points.

When conducting the experiments, the running times of ensemble methods significantly increased the compilation time. This is naturally caused by the numerous sampling of the data and fitting numerous models which creates computational complexity.

7. CONCLUSIONS AND FUTURE WORK

Based on the literature, the state of the art ensemble methods for handling missing data, cover mostly the usage of multiple imputation technique together with another traditional imputation method, or certain statistical algorithms. As previous research has shown, the use of ensemble methods can significantly help in filling the missing data, by combining various techniques. The area of missing data is not broadly explored in terms of machine learning ensemble methods yet, but the discoveries done so far provide great base for future research.

As the experiments suggest, ensemble methods have a significant impact on machine learning algorithms, that can be classified as weak learners. For Decision Tree Regression, they yield model fit score improvement on both small and large dataset used in the study. In measurable terms, as can be seen in **Figure 6.** showing the model performance on a large dataset, the ensemble methods of DT offer an increment of around 10-15%, varying on the method. On the Heart Disease dataset, which is a very small dataset, the accuracy of predictions increase is even more noticeable (at times even more than 40% compared to the base learner, see **Figure 8.**), due to the additional training data generated and models fitted by Boosting and Bagging algorithms.

While in the regression problem the use of ensemble methods give significant score increase, the classification problem is more difficult to clearly judge based on the results obtained. From the selected regression models for this experiment, as earlier mentioned, the largest model fit score improvement can be seen in Decision Tree. The ensemble methods outperform the base algorithm as they are much more robust and limit overfitting by increasing the variance, as well as decrease the error by reducing bias. **Figure 6.** shows that the overall model fit score remains quite stable given various testing data amounts. This is a reasonable result as the training set is large enough and remains unchanged throughout all computations.

While testing the model fit performance on the Heart Disease dataset, we can notice that the performance of base learner regression models is most of the time just as good, or better, than of the ensemble methods. The results of Linear Regression and Bayesian Ridge Regression show that there is almost no difference between the individual ML algorithm and Bagging ensemble. This result is not much of a surprise, as these models are rather stable and creating multiple samples of the data to introduce more diversity does not improve the performance. Both models got a lesser model fit score when trained using AdaBoost.

It is crucial to point out, especially when working on large collections of data, that using ensemble ML models might not be feasible, due to computational complexity. If the used technique does not provide sufficient improvement to the model fitting and predictions accuracy, the trade-off for compilation time and complexity might not be worth it.

7.1 Future work

Hopefully, this paper will inspire further investigation in the field of ML ensemble methods used for handling missing data. Since the scope of this paper did not cover more questions or directions of this research area, there is definitely more work to be done in the future, such as:

- Explore the impact of other ensemble learning techniques such as Weighted Majority Voting, Stacking, and others.
- Experiment with other ML algorithms.
- Compare the performance of handling missing data by ML ensemble methods to traditional missing data imputation techniques, on various dataset sizes.
- Inspect how various ratios of training/testing data affects the model fit score and accuracy of predictions.

8. ACKNOWLEDGMENTS

I would like to thank Dr. Ir. Maurice van Keulen for his support and advice during this research, which allowed me to overcome technical obstacles.

9. REFERENCES

[1] A. C. Acock. Working with missing values. *Journal of Marriage and family*, 67(4):1012–1028, 2005.

[2] D. Bertsimas, C. Pawlowski, and Y. D. Zhuo. From predictive methods to missing data imputation: an optimization approach. *The Journal of Machine Learning Research*, 18(1):7133–7171, 2017.

[3] L. Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[4] G. Chhabra, V. Vashisht, and J. Ranjan. A classifier ensemble machine learning approach to improve efficiency for missing value imputation. In *2018 International Conference on Computing, Power and Communication Technologies (GUCON)*, pages 23–27, 2018.

[5] A. R. T. Donders, G. J. Van Der Heijden, T. Stijnen, and K. G. Moons. A gentle introduction to imputation of missing values. *Journal of clinical epidemiology*, 59(10):1087–1091, 2006.

[6] B. Efron. Missing data, imputation, and the bootstrap. *Journal of the American Statistical Association*, 89(426):463–475, 1994.

[7] C. K. Enders. *Applied missing data analysis*. Guilford press, 2010.

[8] N. J. Horton and S. R. Lipsitz. Multiple imputation in practice: comparison of software packages for regression models with missing variables. *The American Statistician*, 55(3):244–254, 2001.

[9] N. Laranjeiro, S. N. Soydemir, and J. Bernardino. A survey on data quality: classifying poor data. In *2015 IEEE 21st Pacific rim international symposium on dependable computing (PRDC)*, pages 179–188. IEEE, 2015.

[10] X. Lu, J. Si, L. Pan, and Y. Zhao. Imputation of missing data using ensemble algorithms. In *2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, volume 2, pages 1312–1315. IEEE, 2011.

[11] L. Nanni, A. Lumini, and S. Brahnam. A classifier ensemble approach for the missing feature problem. *Artificial intelligence in medicine*, 55(1):37–50, 2012.

[12] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, 11:169–198, 1999.

[13] I. PyCharm Python. for professional developers. *Dosegljivo: https://www. jetbrains. com/pycharm/[Dostopano 2016-08-29]*.

[14] J. R. Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[15] E. Rahm and H. H. Do. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23(4):3–13, 2000.

[16] D. B. Rubin. *Statistical analysis with missing data*. Wiley, 1987.

[17] J. L. Schafer and J. W. Graham. Missing data: our view of the state of the art. *Psychological methods*, 7(2):147, 2002.

[18] C. T. Tran, M. Zhang, P. Andreae, B. Xue, and L. T. Bui. Multiple imputation and ensemble learning for classification with incomplete data. In *Intelligent and Evolutionary Systems*, pages 401–415. Springer, 2017.

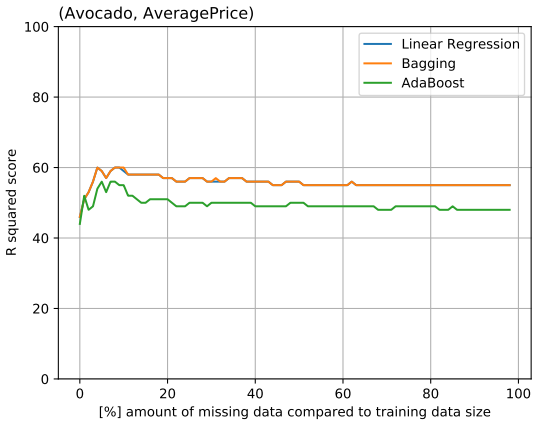
[19] B. Twala and M. Cartwright. Ensemble imputation methods for missing software engineering data. In *11th IEEE International Software Metrics Symposium (METRICS'05)*, pages 10–pp. IEEE, 2005.

[20] B. Twala and M. Cartwright. Ensemble missing data techniques for software effort prediction. *Intelligent Data Analysis*, 14(3):299–331, 2010.

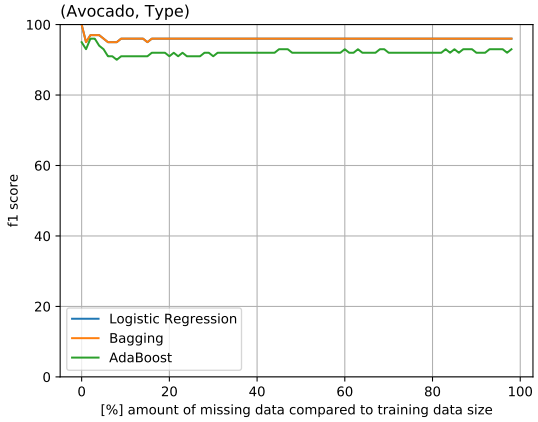
[21] B. Twala, M. Cartwright, and M. Shepperd. Ensemble of missing data techniques to improve software prediction accuracy. In *Proceedings of the 28th international conference on Software engineering*, pages 909–912, 2006.

[22] Z.-H. Zhou. *Ensemble methods: foundations and algorithms*. CRC press, 2012.

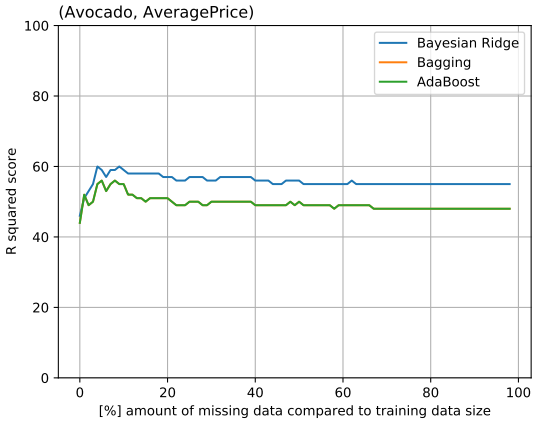
APPENDIX



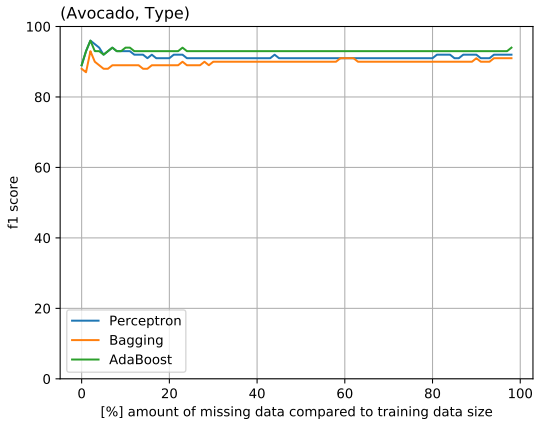
(a) *Linear Regression and its ensembles on 'Average Price' attribute*



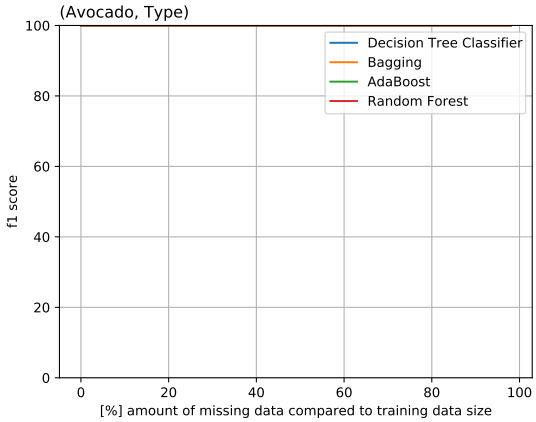
(b) *Logistic Regression and its ensembles on 'Type' attribute*



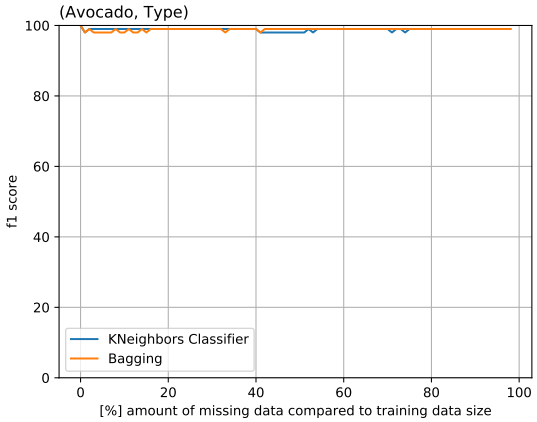
(c) *Bayesian Ridge Regression and its ensembles on 'Average Price' attribute*



(d) *Perceptron and its ensembles on 'Type' attribute*

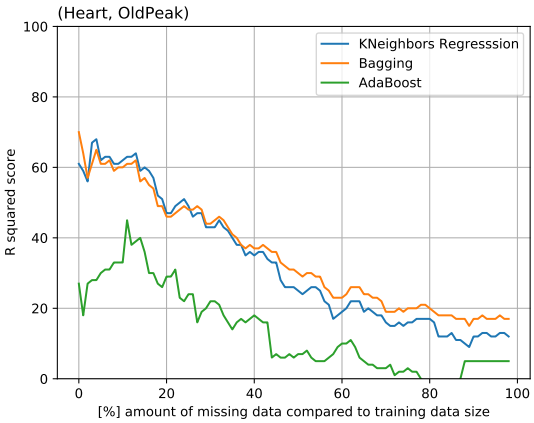


(e) *Decision Tree Classifier and its ensembles on 'Type' attribute*

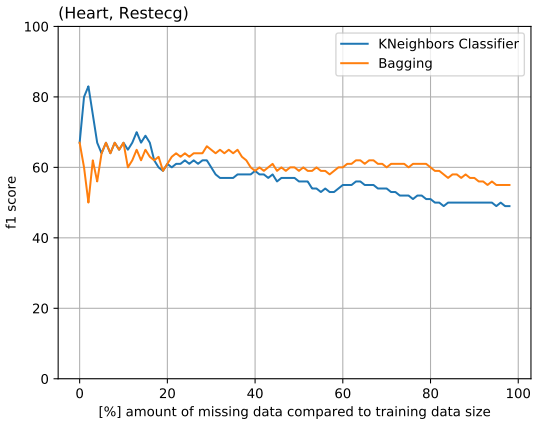


(f) *KN Neigbors Classifier and its ensembles on 'Type' attribute*

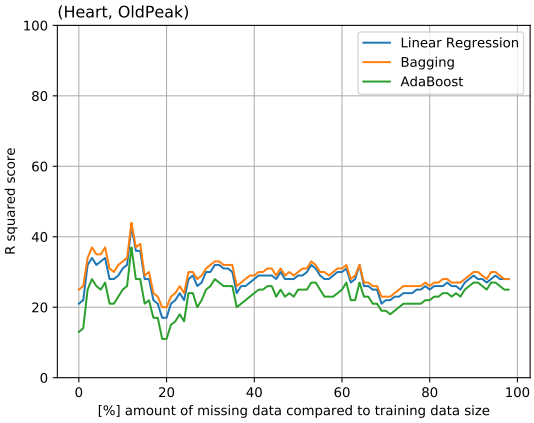
Figure 10: Score obtained on the training set from Avocado dataset, per base model.



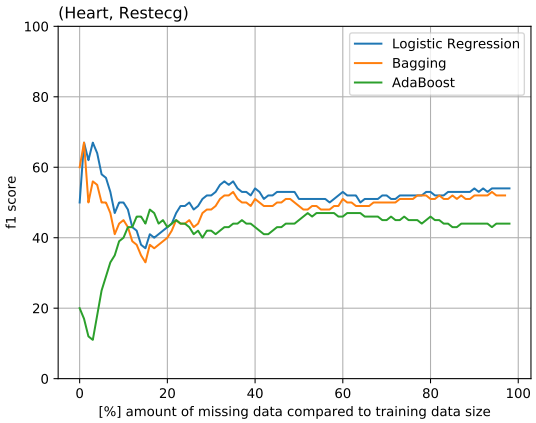
(a) *KN Neigbors Regression and its ensembles on 'Old Peak' attribute*



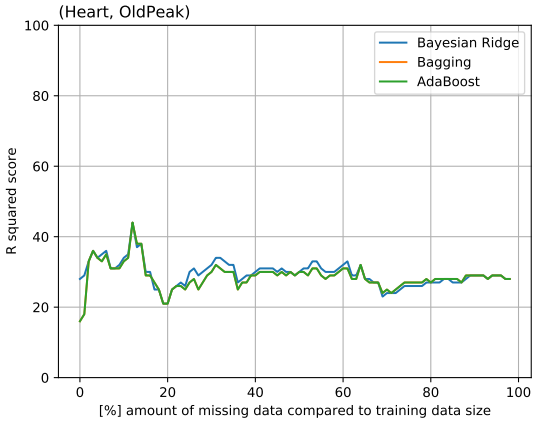
(b) *KN Neigbors Classifier and its ensembles on 'Restecg' attribute*



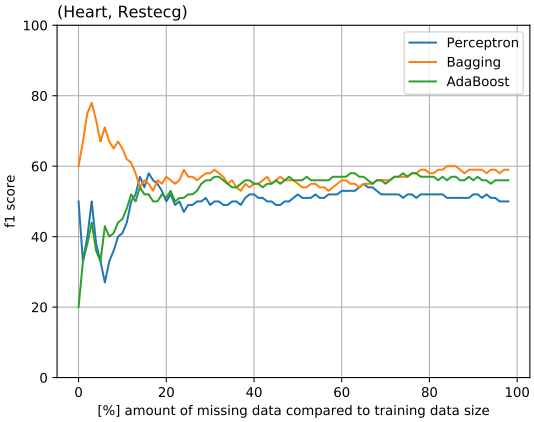
(c) *Linear Regression and its ensembles on 'Old Peak' attribute*



(d) *Logistic Regression and its ensembles on 'Restecg' attribute*



(e) *Bayesian Ridge Regression and its ensembles on 'Old Peak' attribute*



(f) *Perceptron and its ensembles on 'Restecg' attribute*

Figure 11: Score obtained on the training set from Heart Disease database, per base model.