Highlights

SaNDA: a Small and iNcomplete Dataset Analyser

Alfredo Ibias*, Varun Ravi Varma, Karol Capała, Luca Gherardini, Jose Sousa

- Personalised Health datasets are small and full of missing data.
- Traditional methods require huge and complete datasets.
- We propose an explainable method tweaked for those datasets.
- We compare our method with Random Forest and gcForest.
- Our method beats Random Forest when data is missing.

SaNDA: a Small and iNcomplete Dataset Analyser

Alfredo Ibias*, Varun Ravi Varma, Karol Capała, Luca Gherardini, Jose Sousa

[a]*Personal Health Data Science group, Sano - Centre for Computational Personalised Medicine, Czarnowiejska 36 building C5, Kraków, 30-054, Małopolska, Poland*
{a.ibias, v.varma, k.capala, l.gherardini, j.sousa}@sanoscience.org*
**Corresponding Author*

Abstract

In personalised health, small datasets with missing data are quite common. Current Machine Learning methods are unable to process such datasets in a meaningful way due to the huge data volume requirement. To address this problem, we propose a new Small and iNcomplete Dataset Analyser (SaNDA) to process such datasets in a meaningful way. Due to the characteristics of these datasets and the criticality of the domain, an explainable method is mandatory to facilitate decision-making interpretation. Thus, SaNDA prioritises explainability over efficiency by design. We evaluated our proposal against Random Forest as a baseline for explainable methods, and against gcForest as state-of-the-art for small datasets. We observed that our proposal outperforms Random Forest when there is more missing data and/or lower number of entries in the dataset, obtaining less favourable results over larger, well-curated datasets. It is also preferable than gcForest due to its explainability and privacy protection capabilities. Given the difficulties in obtaining complete, reliable data in the healthcare field, we consider that our proposal could be useful for practitioners.

Keywords: Personalised Health, Small Data, Machine Learning, Data Science

1. Introduction

In personalised health, small and incomplete datasets are the norm, in the form of what has been termed "small data" [18]. Due to the nature of the studies performed, recording data is an expensive and laborious task, often prone to errors. The daily work of a clinician is a balance between time and

accuracy of the assessment, whereas data collection for secondary use is not a central concern. Moreover, the data is usually collected with the aim to improve a single unit such as city, hospital, or healthcare system, thus having a limited scope [18]. This makes the collection of data an error-prone process, where on the one hand the data collected usually integrates a semantic bias coming from variations in context, uncertainties in measurements, and from basic intuition utilised during diagnosis; and on the other hand inconsistencies arise from the impossibility of the available disease classification protocol to accommodate all the possibilities (ICD9/ICD10) [5, 33], and from information incompleteness produced by the mobility of the individuals within the healthcare system without clear tracking. This results in difficulties generating big, clean and well-curated data sets. To facilitate overcoming these limitations, a definition of the minimal or standard content to be recorded would be useful. However, such definition has not been set yet, due to lack of agreement on the amount of information needed.

Traditional Machine Learning (ML) methods under-perform when provided with this kind of datasets [11]. Due to the statistical nature of learning algorithms, and the required generalisation techniques, a huge amount of data is necessary to obtain satisfactory results with these methods. Moreover, these methods usually rely on clean and complete datasets, filling or removing empty entries. These requirements make the use of ML methods to analyse small and/or incomplete datasets almost pointless, as the results would be of limited use.

Moreover, several crucial fields started demanding insights and explanations about the inner decision-making process of automatic systems, such as finance, privacy, law, and healthcare [2, 3, 29, 37]. This rising concern resulted in the recital 71 of the General Data Protection Regulation (GDPR) of the European Union and in the Algorithmic Accountability Act decreed by the US congress (H.R. 6580 of 2022) [22, 12]. A new current, named eXplainable AI (XAI), acknowledged this necessity and formulated its principles in terms of transparency, interpretability, and explainability [3]. Providing a system able to explain its inner functioning can be exploited to ensure fairness, remove or limit biases, and improve the performance by changing the inference process. Building transparent models can lead to an inferior performance than current learning techniques, but these perks are more valuable in delicate contexts, and can be improved to the state-of-the-art performance in the long run. Understanding the functioning of an automatic decision process is also essential to become trustworthy for its final users, such as

clinicians and patients. Thus, traditional ML methods, and especially Deep Learning methods, are not useful in this context due to their opaque nature, often referred as "black box". The only widely accepted method that fills this requirement is Decision Trees, and its improved version, Random Forest. However, these methods cannot deal with missing data. Nevertheless, we will use them as a baseline to compare our proposal with, since they have properties we would require in any system designed for tackling medical data.

All these factors result in plenty of incomplete, small datasets being produced in the healthcare field that cannot benefit directly from classical ML methods. Thus, there is a need for novel methods specially designed to deal with such datasets. In the field, it is assumed that representation of the knowledge emerging from data, suited for self-supervised learning would be richer than those generated by supervised learning using labeled data [16, 32, 38]. In this work we present our first approach to explore emergent knowledge mined from available data. Specifically, we aim to develop a tool able to analyse and produce meaningful results from small, incomplete datasets, while at the same time being explainable and GDPR compliant. Our proposal is a Small and iNcomplete Dataset Analyser we call SaNDA. It comprises a statistically defined stratification mechanism, allowing for a better understanding of the relevant features that define the classes in which we split the data. It integrates three phases: *data abstraction*, to preserve data privacy while allowing better decision making on complex scenarios [19]; *knowledge graph generation*, to provide explainability through knowledge graphs [21, 8]; and *classification*, to validate the generated knowledge graphs [35].

To evaluate the effectiveness of our proposal in analysing reality, we devised some experiments comparing it with Random Forest - the explainable ML method par excellence -, and gcForest - the state-of-the-art for dealing with small datasets. We compared their capability of performing a binary classification task, over small and medium size datasets with different levels of missing data. The results show that our proposal beats Random Forest when the level of missing data increases, or when the size of the dataset gets smaller. They also show that our proposal is better suited for dealing with missing data and imbalanced datasets. This shows that our proposal achieves its goals, being better fitted for small, incomplete datasets, like the ones we can find in the healthcare research field.

Additionally, SaNDA provides a graph-like visualization of the learned data, as well as the raw numbers of such graph. As a tool, it would allow

any practitioner to qualitatively (and quantitatively) analyse the relevance of different features for the binary classification task at hand. This could be very useful for practitioners that need to know which features of their dataset are more relevant for determining the class of a sample, what is something most ML methods cannot provide in a human understandable way.

The rest of the paper is organised as follows: In Section 2 we briefly introduce previous work related to our proposal. In Section 3 we introduce two basic concepts essential for understanding the work done in this paper. In Section 4 we explain our proposal in detail. In Section 5 we present the experiments performed to validate our proposal. In Section 6 we explain some decisions taken during the development of our proposal. In Section 7 we discuss the threats to the validity of our results. Finally, in Section 8 we present the conclusions of our work.

2. Related Work

From its inception up to the actual advent of big data and computational power, a fundamental premise behind Artificial Intelligence (AI) has been its inspiration from natural human intelligence. Achieving common sense and models of thought, and developing computational approaches able to simulate human behaviour were thought to be fundamental elements to achieve what is now termed Artificial General Intelligence (AGI) [25]. Despite its ambitious inception, with the advent of the perfect storm of availability of computational power and increased access to big data, the focus changed to a brute-force approach on data, deviating from human-inspired machine intelligence. This line of thought resulted in a generalised idea that, to achieve machine intelligence comparable with humans, scale is all that is necessary [26].

However, this trend left behind scenarios where the access to huge amount of data is not viable. Thus, work on analysing and processing small datasets, especially small and incomplete datasets, is scarce. Moreover, this kind of work usually is not connected to the current trend in AI and can not take advantage of its advancements, like the ones produced in deep learning or reinforcement learning.

The few works that deal with small datasets usually involve using transfer learning from models trained with huge datasets [40] or developing handcrafted methodologies to solve specific problems [41]. There are also few works that try to adapt classical deep learning methods to learn from small

datasets [34], but their effectiveness is limited. The closest architecture that uses non-differentiable modules is Deep Forests [42], which use Cascading Forest architectures and Multi-Grained Scanning to account for layer-based processing, feature transformation and complexity. Similar tree-based architectures are discussed in [14, 7], with specific changes to target problems of high-dimensionality and class imbalance.

However, in neither of these cases the proposed method is easily explainable and/or able to deal with missing values, as we aim to achieve with our proposal. To the best of our knowledge, our work is the first that tries to tackle this specific scenario, where we have small datasets with missing values.

3. Theoretical Background

To understand our work first we need to introduce some basic concepts we use in it, which are based in previous research. Specifically, we need to present the concepts of Receiver Operating Characteristic curve theory, Random Forest, and Cascading Forest.

3.1. Receiver Operating Characteristic curve theory

The Receiver Operating Characteristic (ROC) curve is a statistical methodology to analyse binary classifiers. Given a binary classification task, ROC curve theory defines one of the classes as *positive* and the other one as *negative*. Based on the results of the classification task, this theory defines the following basic concepts:

- True positives (TP): number of correctly classified samples of the positive class.
- False positives (FP): number of incorrectly classified samples of the negative class.
- True negatives (TN): number of correctly classified samples of the negative class.
- False negatives (FN): number of incorrectly classified samples of the positive class.

With these concepts, it is possible to define different formulas to explore the results of a binary classifier. In our case, we are interested only in the balanced accuracy (BA) formula (Eq. 1), that measures the accuracy of the classifier, even under the case of a class imbalance in the dataset.

$$BA = \frac{\frac{TP}{TP+FN} + \frac{TN}{TN+FP}}{2} \tag{1}$$

3.2. Random Forest

Random Forest [6] is one of the simplest Decision Tree based ensemble learning methods for classification. During training, a multitude of trees are constructed, and the class label is decided on the basis of the votes of each tree. Decision Trees use the concept of entropy [23] to create splits or decision boundaries for each feature in the data; the choice of feature is dependent on the Maximal Entropy Principle [13]. We chose Random Forest as our comparison method because it is widely used (especially in industry) as an explainable method, and performs well on average.

3.3. Cascading Forest

Cascading Forest architectures [42, 14, 7] utilise *ensemble of ensembles* methodology to perform similarly to the layered nature of neural networks, using forests as alternatives to differentiable neurons. This allows the combination of trees to capture more complex conditions. The cascade also uses boosting, collating the class votes from previous layers into inputs for succeeding layers. Cascading forests allow a small level of interpretability of decision boundaries, since they potentially scale up to large number of decision forests leading to large decision diagrams, depending on the hyperparameters set by the user or the optimisation metric chosen. Of the multiple available architectures, we chose gcForest [42] due to its simplicity, availability of code, and design motivation, which are similar to ours. Variations of the gcForest algorithm have also been used for image classification and modelling genetic data.

4. SaNDA: a Small and iNcomplete Dataset Analyser

As explained before, our proposal faces a very specific scenario: a small size dataset with some level of missing values and a binary classification task. Additionally, it is required to be explainable and GDPR compliant. Furthermore, its representation has to be validated to ensure it is correct. With

these requirements, we developed a three-stages algorithm: preprocessing, knowledge graph generation, and classification.

In the preprocessing stage, we introduce measures for privacy protection and data analysis, as well as a way of dealing with the missing data. We then transform the preprocessed data into a graph-based representation of the knowledge comprised in the dataset, extracting the relevant information in an explainable way. Finally, the classification stage is where the graph model is validated, classifying the values between the available classes and obtaining an accuracy measure that represents how well the method has learnt the patterns in the input data.

The whole process is shown in Figure 1, where each stage has a different color: the blue blocks compose the preprocessing stage, the green ones build the graph generation stage, and the orange blocks compose the classification stage. The knowledge graph generation stage is further divided between the generation of the *Seed Models* (dark green) and of the *Statistically Augmented Models* (light green), division that we will explain later.

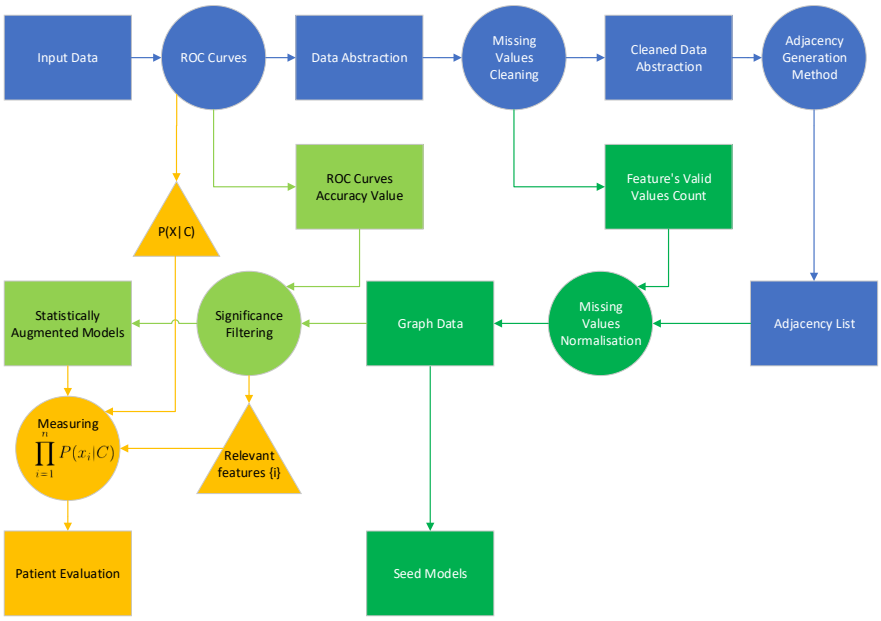

Figure 1: Process pipeline: SaNDA

4.1. Preprocessing

In the preprocessing stage we take the input data and generate an abstraction of each feature of the dataset. These abstractions allow us to better normalise the data, as well as to anonymise it. Anonymising the data would be fundamental to comply with regulations about data protection, such as GDPR. We define an abstraction as a mapping of the dataset values (origin set) to a finite set of values with a smaller cardinality (target set).

Definition 1. *Given A and B two sets of numerical values, with $|A| \geq |B|$, we define an abstraction as a function $\mathcal{F} : A \to B$ that maps the values of A to values of B.*

Although the size of the target set could be any natural number (as long as it is smaller than the size of the origin set), in our case we will explore the case of a binary set (that is, a set with two values). This decision allows us to experiment with the family of abstractions that can produce the highest loss of information. Therefore, any abstraction to a target set with more possible values should get better results, although exploring it would be matter of future work. In our specific case, the target set is going to be the set $\{UP, DOWN\}$.

There are multiple possibilities to generate an abstraction, including abstracting to the class labels. However, we would like our abstractions to keep the information of the raw data as well as maximise the information available for binary classification. As we are working with numerical data to be transformed into binary values, we only consider the abstractions that divide our data into two sets based on a specific *cut-off value*. Of the many alternatives to compute cut-off values, we propose the use of ROC curve theory, and specifically the use of balanced accuracy (Eq. 1). This is motivated by the fact that having a higher balanced accuracy means that the generated abstraction has a bigger amount of information to discriminate between the two classes, while at the same time it accounts for imbalanced datasets. Thus, we look for the cut-off value that produces the abstraction with the highest balanced accuracy. This way, we will have the set of values greater than the cut-off value (UP) and the set of values lesser than or equal to the cut-off value ($DOWN$), which results in our binary abstractions.

Formally, the abstracted value of an element is computed using the following formula:

$$\mathcal{F}(x) = \begin{cases} UP & \text{if } x > x' \\ DOWN & \text{if } x \leq x' \end{cases}$$

where x' is the cut-off value. This value fulfils the following formula:

$$x' \in \{x | BA(x) \geq BA(y)\ \forall y \in A\}$$

where A is the set of values to abstract and $BA(x)$ is the balanced accuracy obtained by splitting the set A by the cut-off value x.

As explained before in Section 3, we use balanced accuracy (Eq. 1) instead of the classical concept of accuracy, because balanced accuracy will not be biased by imbalances in the classes composing the dataset.

Notice that this abstraction is performed independently for each feature in the dataset. This is a crucial point, since using the same abstraction for different features will produce undesired effects, especially if different features have different orders of magnitude. This fact also helps dealing with small datasets, as we transform a myriad of different values into two values that keep repeating, thus projecting the data into a discrete space. This makes modelling the underlying probability distribution easier, as we need fewer datapoints to represent it.

Finally, it is important to remark that this methodology is a first approach to generate meaningful abstractions. We based our choice on the ROC curve theory, as it is commonly used in medical studies to evaluate classifiers [24, 15, 20]. However, further research into developing better abstraction generation approaches would be required, especially for the case where we have more than two classes. For now, we consider this useful enough for our proposal, for the scenario we are exploring.

After computing the abstraction of each feature, the cut-off values and the balanced accuracy obtained for each feature are recorded. The former will be useful to enhance the explainability, as an abstraction of a feature to *UP* means that the value is higher than the cut-off value and *DOWN* means that it is lower; and the latter will be used for analysis in later stages to determine which features are more relevant for the classification and which ones are not so relevant, as we can observe in Figure 1.

Once we abstracted the data, we segment it into different groups based on the class each one belongs to. In this step we compute and store the probability of each feature being in each state (*UP* or *DOWN*) given the class of the group analysed. Thus, for each feature, for each class, we get a different probability for each possible state. These probabilities are the fundamental way we deal with small datasets: as we have fewer datapoints to learn from, we compute the underlying probability distribution directly from

them, rather than approximating it in an iterative manner. These probabilities will be later on used to evaluate our graph knowledge representation, as we can observe in Figure 1.

After this segmentation step, we address the problem of missing or null data. To this end, we replace all the null and missing values for a *null* token value, and we store the count of valid values per feature per class, so we end up with a different count for each class. These counts will be later used for normalising values when generating our knowledge graph representation, as we can observe in Figure 1.

The last step is fundamental to deal with incomplete datasets. In classical Machine Learning, having missing data leads to having to either replace missing values or discard the entire record, therefore creating fictitious data or removing useful information. However, these options usually lead to biases and worse performance, as they are either creating data or discarding data. In both cases, these modifications lead to modifying the underlying probability distributions, thus modifying the data from which the method learns. In our approach, however, we do not create neither discard data, as we just map the underlying probability distribution with the available data. The fact that we do this in a per feature basis allows us to not worry if an entry has missing data for other features, as we analyse each feature separately.

Finally, the last step of our preprocessing stage is transforming these abstracted and cleaned datasets (one per class) into adjacency files, where each row is a relationship between two features in their corresponding states. To this end, we generate an adjacency for each feature of a entry of the dataset with any other feature of the same entry, skipping those that have the *null* token value. After this step, the null values are no longer considered, as they are being ruled out of the final adjacency file. We have to take special care of not repeating the adjacencies in the inverse direction, as the graphs we are going to generate are not directed. This way we get, for the same feature in the same state, as many adjacencies as it appears in the dataset, multiplied by the number of other features. In the end, we would get as many adjacency files as classes, which would allow us to generate a different knowledge graph representation per class, enabling comparisons.

4.2. Graph Generation

Once we have an abstracted and clean version of our data in a convenient adjacency file, we generate an explainable knowledge graph representation,

following the work of [21]. For our purpose, we combine the classical definition of a weighted graph with the definition of an additional vertex property:

Definition 2. *A graph is a set* $G = \{V, E\}$ *where* $V \in \mathbb{N}$ *is a set of vertices and* $E \in \mathbb{N} \times \mathbb{N}$ *is a set of edges connecting two vertices. A weighted graph additionally has a map* $W : E \to \mathbb{R}^+ \cup \{0\}$ *that maps each edge to a weight. Each vertex has a significance defined by a map* $S : V \to \mathbb{R}^+ \cup \{0\}$*.*

To create these graphs, we load the adjacency files of each class. We create a node for each feature state in the adjacency file, and the number of times two feature states appeared together represents the strength of their edge, normalised by the amount of valid values of both features. Through this procedure, we build a representation of the class using the available features. As we have two counts of valid values, one for each feature forming the edge, we use the smaller value, as the maximum possible number of adjacencies is determined by the feature that has the lowest number of elements. Finally, the vertex significances will be the probabilities of each feature being in each state (*UP* or *DOWN*) given the class, previously stored in the preprocessing stage. We call these graphs Seed Models (SMs), as they represent the whole dataset.

After generating the graphs, we also compute the similarity score [30] between the stratified classes, and the central point dominance [10] and entropy [28] of each class. Additionally, we plot them into a `.tiff` file where we can explore them visually, and store in text files the edge weights and the vertex significances for easy access.

In the second step, we identify the features of low importance for the classification task. We consider as irrelevant features those having a low ROC curve accuracy. We use the error margin to distinguish between low and high-accuracy features.

Definition 3. *Given a sample size* n *and a classification result* p*, and that to achieve a confidence of* 99% *we need to use a* $z - value$ *of* 2.58*, we define the error margin produced in the classification as:*

$$2.58 \cdot \sqrt{\frac{p \cdot (1-p)}{n}}$$

We compute this error margin for each feature using the ROC curve balanced accuracy obtained in the preprocessing stage as p and the size of

the dataset as n. Then, if the balanced accuracy is under 50% plus the computed error margin, the feature is considered irrelevant.

Once we have a set of relevant features, we can produce new graphs using only them and filtering out the irrelevant ones. Thus, we produce the Statistically Augmented Models (SAM), that are a reduced version of the original data, including only the relevant information for differentiating between classes. The graph generation process is identical to the Seed Model generation except for the adjacencies including irrelevant features, which are skipped. For these graphs, we also compute the similarity score between the stratified classes, and the central point dominance and entropy of each of the classes. As before, we plot them into a `.tiff` file, and store the edge weights and the vertex significances into text files.

The visual graph representation enables better exploration of the data structures and of their relationships across the feature states. We can also observe how different feature states evolve across classes, and to check these results with the actual values we have the files storing the edge weights and vertex significances. An example of the difference between a SAM model and its respective SM model can be found in Figure 2, where we display the SM for class 0 and class 1 in the top, and the SAM for those classes in the bottom. Significant differences in the structure of the graph are visible, from the number of nodes to the generated communities.

This stage is essential when dealing with small datasets. As we have fewer datapoints, being able to explore the underlying probability distribution of each of their features, and the relative relevance of each feature for the final classification, allows us to get more information about how to properly analyse such datapoints. It also allows us to explore the information arising from the dataset in a way no other method has provided previously. Thus, even if we have very few datapoints, we still can get some knowledge out of them.

4.3. Classification

Finally, we aim to classify the entries of the dataset using the generated graphs. This would allow us to validate that our graphs truly represent the knowledge comprised in the dataset. To this end, we devised a classification method that takes, for each class, the vertex significances of the states of each feature present in the entry and multiply them together, and the class that obtains a higher value is assigned to the entry. This implies that, if an entry has empty or null values, these values are not considered for computing the classification and their corresponding vertex are ignored in the formula.

Remember that each vertex significance is the probability of its associated feature being in a specific state (UP or $DOWN$), given the class that the graph represents. Thus, in the classification we are computing the probability of being in a specific combination of feature states given each class, and we assign the class with the highest probability.

In a formal way, given an entry $X = \{x_1, \cdots, x_n\}$, for each class C we are computing the following probability:

$$P(X|C) = \prod_{i=1}^{n} P(x_i|C) \tag{2}$$

Then, for the binary classification task, given two classes C_1 and C_2 and an entry $X = \{x_1, \cdots, x_n\}$, we can compute the following ratio:

$$\mathcal{P}_X = \frac{P(X|C_1)}{P(X|C_2)} = \prod_{i=1}^{n} \left(\frac{P(x_i|C_1)}{P(x_i|C_2)} \right) \tag{3}$$

where if $\mathcal{P}_X > 1$ then the entry is more likely to be in the C_1 state, otherwise it is likely to be in C_2. For the multi-class case, we would just take the class with the higher $P(X|C)$.

With this method, we get a different classification for SMs and SAMs. For classification purposes we will use the SAM since it filters out irrelevant features.

Finally, given a classification, we compute both its accuracy and its balanced accuracy (Eq. 1). We will use these values to compare our proposal to other methods, like Random Forest.

5. Experimental Evaluation

To evaluate how well our proposal performs over small to medium size datasets with missing or null values, we developed a comprehensive experiment. To further strengthen our results, we included a comparison with Random Forest and gcForest algorithms.

5.1. Experimental Subjects

First, let us introduce the experimental subjects. We took six datasets which we divided into two groups based on size: small datasets with less than or equal to 10^5 datapoints, and medium datasets with between 10^5 and 10^8

Subject	Size	# features	# entries
small size			
Ionosphere	$11,934$	34	351
Sonar	$12,480$	60	208
Wisconsin Breast Cancer	$17,070$	30	569
medium size			
Accelerometer	$408,000$	4	$102,000$
HIGGS	$2,800,000$	28	$100,000$
Air Pressure System Failure	$10,200,000$	170	$60,000$

Table 1: Characteristics of the experimental subjects.

datapoints. Their characteristics can be found in Table 1. All these datasets are for binary classification.

Following the order of Table 1, the first small dataset is the Ionosphere dataset [9, 36], which contains 351 measures from the radar analysis of the ionosphere from Goose Bay, Labrador. It contains 34 features measuring the number of free electrons in the ionosphere and other electromagnetic signals from it, all in numeric format. The classification task is to determine whether the ionosphere has any kind of structure or not.

The second dataset is the Sonar dataset [9, 1], which contains 208 measures of sonar signals utilised to differentiate between rock and metal surfaces underwater. It contains 60 features measuring the shape and characteristics of the sonar signal, all in numeric format. The classification task is to determine whether the signal bounced off a metal cylinder or off a roughly cylindrical rock.

The third dataset is the Wisconsin Breast Cancer dataset [9, 39], which contains 569 measures from Fine Needle Aspirates of breast mass. It contains 30 features measuring shape and composition of a breast mass, all in numeric format. The classification task is to determine whether a breast mass is cancerous or not.

For the medium size datasets, the first one is the Accelerometer Data Set [9, 31], containing $102,000$ measures about accelerometer data from vibrations of a cooler fan with weights on its blades. It contains 4 features measuring the vibrations in the 3D space, as well as the velocity of the fan, all in numeric format. The classification task is to determine the orientation of the weights in the blades: either in opposite blades (180 deg) or in neighbouring blades (at 90 deg).

The second dataset is the HIGGS Data Set [9, 4], which contains $100,000$ measures about particle decay measurements that could contain the Higgs bosson or not. The data contains kinematic properties of 28 different energy particle measurements, all in numeric format. The classification task is to discriminate between signal processes where a Higgs boson was created and background processes where there were not.

Finally, the last medium size data set is the Scania Trucks Air Pressure System (APS) Failure Data Set [9], which contains $60,000$ measures on APS failure in trucks. It contains 170 features measuring different symptoms of failure in Scania trucks, all in numeric format. As this was an industrial dataset, the features are anonymised. It is also the only dataset having some missing values. The classification task is to discriminate between failures coming from the Air Pressure System (APS) and other failures.

5.2. Experimental Setup

The experimental setup consisted of processing these datasets with both SaNDA and a Random Forest, and performing a classification of all the entries. Then, we computed and compared for both models the accuracy and balanced accuracy (Eq. 1). We repeated this process using the original datasets, as well as versions with added missing data. In the case of Random Forest, in order to be able to process the datasets with missing data, we replace any missing data with 0, as the alternative (erasing the entries with missing data) would lead to a very small, if not empty, dataset.

To artificially introduce missing data we erased a percentage of values of the dataset and transformed them into empty values. We did this process erasing $1\%, 5\%, 10\%$ and 50% of data values. Then we processed these new datasets with both methods and obtained the results displayed in Table 2 for SaNDA and in Table 3 for the Random Forest. In those tables, the winning method has its values in bold. A graphical visualisation of the results can be found on Figure 3, where the results of each dataset are displayed together.

As we can observe from the results, the performance depends a lot on the dataset and the level of missing values. However, there seems to be a trend: in smaller datasets, SaNDA is better. Also, in datasets with more missing values, SaNDA is better too, at least if we look at balanced accuracies. Moreover, in general, SaNDA is better than Random Forest the 60% of the time (if we consider balanced accuracy). Thus, these results reinforce the usefulness of our proposal for the specific scenario we wanted to address. However, from these results we can also observe that for medium size datasets, or datasets

with no missing values, SaNDA is not a better option than Random Forest. Expanding SaNDA to work also in currently unfavourable scenarios is a concern to address in future work.

There is an additional interesting conclusion: SaNDA works better when there is more class imbalance. This can be observed by the fact that SaNDA beats Random Forest in the datasets that have class imbalance (except the HIGGS dataset), but also by the fact that it beats Random Forest only in balanced accuracy for the Air Pressure System Failure dataset.

5.3. Additional Experiment

As an additional experiment, we wanted to explore how SaNDA would compare with state-of-the-art algorithms focused on dealing with small datasets. Currently, the best alternative for small datasets is gcForest [42] as discussed in Section 3. This algorithm is based on a ensemble of forests and it is focused on dealing with small datasets. However, this algorithm focuses on efficiency over explainability, so the latter is lower than that of a Random Forest, while its performance is improved. Thus, we expected that it will get better results than our proposal, but we wanted to explore how much better.

To compare our proposal against gcForest we decided to use the same experimental setup than in the previous experiment, exchanging Random Forest with the gcForest alternative. To compute gcForest, we used the version that is publicly available courtesy of the authors. The results of the experiments can be found on Figure 4, where the results of each dataset are displayed together.

As we can observe, our proposal is worse than gcForest in almost any scenario except for the Air Pressure System Failure dataset. In this case, gcForest obtains better accuracy, but the balanced accuracy is worse than both the accuracy and balanced accuracy obtained by our proposal. This reinforces the idea that SaNDA works better when there is more class imbalance. Another case to analyse is the Sonar dataset, where a huge amount of missing values leads to SaNDA performing better than gcForest. This reinforces the idea that SaNDA works better when there are more missing data. In conclusion, having these results, and the fact that SaNDA is more explainable and privacy safe than gcForest, we conclude that our proposal is valuable for dealing with the specific scenario we are exploring in this paper.

6. Discussion

In this section we want to discuss some decisions taken and some characteristics of our proposal. Specifically, we want to talk about why we used a probability as the vertex significance, and why we do not have a division between training and testing.

Regarding the computation of the significance of the vertices, a clear concern we had during development was that our solution did not include any knowledge about the graph structure in the vertex significance. To solve this concern, we actually explored multiple alternatives (like using the famous PageRank algorithm [17]). However, amongst the alternatives we tried, the only one that produced results closer to the results obtained by our proposal was the use of PageRank with a directed graph in which the edges contained the conditional probability of having the target feature state given the origin feature state. However, the results in our main experiment where not promising at all, obtaining worse results than our proposal for most of our datasets.

To be precise, we obtained the results displayed in Figure 5, in which we observe that both directed PageRank and probabilities beat the other one half of the time. However, while using probabilities delivered better results compared to Random Forest (beating it 60% of the time), directed PageRank delivered worse results compared to Random Forest (beating it only 36.66% of the time).

An important remark here is that, given these results, for some datasets the directed PageRank version worked better. Moreover, when testing with our internal datasets, we observed a similar situation, where for some datasets, this approach was better than our proposal. However, there is not enough data to conclude that this was a better option than our proposal. Thus, further work should be done to explore why, for some datasets, directed PageRank alternative delivers better results.

Regarding the fact that we do not split our dataset into a set for training and other for testing, we want to stress that the goal of our proposal is not to classify individuals. Moreover, our proposal does not have any generalisation mechanism. Thus, the goal of our classification phase is not to properly classify new individuals, but to give a level of confidence on the representativeness of the generated graph. All of this implies that we do not need to assess the generalisation capabilities of our proposal with a testing phase. Moreover, we actually want to assess how well we have over-fitted to

our dataset. Thus, the classification phase is done with the same datapoints as the ones used for training.

7. Threats to Validity

In this section we discuss the possible threats to the validity of our results. The first kind of threats we consider are the threats to internal validity, that can explain our results due to uncontrolled factors. The main threat in this category is the possibility of having faulty code. To reduce this threat we have carefully tested each piece of code used in our experiments and we have relied on widely tested libraries like scikit-learn [27] for the Random Forest algorithm and the authors implementation for gcForest. Another threat in this category is the impact of randomisation in our results. To control this factor we have controlled the random seeds when needed, for both reproducibility and comparison purposes.

The second kind of threats are the threats to external validity that hamper the generality of our results to other scenarios. In our case the only threat in this category is the small scale experimental setup, having tested our proposal over six datasets. However, we have performed small experiments with other internal datasets too, and obtained similar results. Nonetheless, the exploration of the performance of our proposal in other scenarios remains part of future work.

Finally, the last kind of threats are the threats to construction validity, hampering the extrapolation of our results to real-world scenarios. In this case, the range of possible scenarios is potentially infinite, and this threat cannot be fully addressed, but as explained before, the exploration of the performance of our proposal in other scenarios is matter of future work.

8. Conclusions

Small, incomplete datasets are quite common in healthcare research. This hampers the application of traditional Machine Learning (ML) methods due to their need for huge amounts of clean data. Moreover, classical ML methods tend to not be very explainable, what is a basic requirement in a critical environment like healthcare. Thus, the development of a method to address this kind of scenarios is fundamental. To that end, we have proposed SaNDA, a Small and iNcomplete Dataset Analiser, able to deal with small, incomplete datasets, and at the same time being explainable and GDPR compliant.

Our proposal has three stages: preprocessing, to prepare the data and protect privacy; knowledge graph generation, to provide a graphical and explainable knowledge representation; and classification, to validate the knowledge comprise in the graph knowledge representation. To evaluate our proposal we tested it against the Random Forest and gcForest algorithms over six small to medium size datasets. The results show that SaNDA performs better the smaller and the more incomplete a dataset is, and it has a clear advantage over imbalanced datasets.

For future work, we would like to explore how to extend SaNDA to beat Random Forest and gcForest also in medium size datasets, and/or in clean curated datasets. We would also like to explore alternative abstraction generation methods, specially those able to abstract to more than two classes. We would like to explore what is the performance of SaNDA in other scenarios too. Finally, we would like to explore alternative methods for the graph generation, in order to improve SaNDA explainable capabilities.

Acknowledgments

We would like to thank our colleagues Anna Drożdż for her idea of using ROC curves as a classifier, and her and Kamil Woźniak for their useful comments, discussions and testing of the tool. This research has been supported by the European Union's Horizon 2020 research and innovation programme under grant agreement Sano No 857533. This publication is supported by Sano project carried out within the International Research Agendas programme of the Foundation for Polish Science, co-financed by the European Union under the European Regional Development Fund. This research was supported in part by PLGrid Infrastructure.

References

[1] Connectionist Bench (Sonar, Mines vs. Rocks). UCI Machine Learning Repository.

[2] Plamen Angelov and Eduardo Soares. Towards explainable deep neural networks (xDNN). *Neural Networks*, 130:185–194, 2020.

[3] Plamen P. Angelov, Eduardo A. Soares, Richard Jiang, Nicholas I. Arnold, and Peter M. Atkinson. Explainable artificial intelligence: an

analytical review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 11(5):1–13, 2021.

[4] P. Baldi, P. Sadowski, and D. Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5(1):1–9, 2014.

[5] T. Botsis, G. Hartvigsen, F. Chen, and C. Weng. Secondary Use of EHR: Data Quality Issues and Informatics Opportunities. *Summit on translational bioinformatics*, 2010:1–5, 2010.

[6] L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, 2001.

[7] Hao Chen, Chaoshun Li, Wenxian Yang, Jie Liu, Xueli An, and Yujie Zhao. Deep balanced cascade forest: An novel fault diagnosis method for data imbalance. *ISA transactions*, 126:428–439, 2022.

[8] F. Diaz, C. Shah, T. Suel, P. Castells, R. Jones, T. Sakai, J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, and X. Xie. Self-supervised Graph Learning for Recommendation. *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 726–735, 2021.

[9] D. Dua and C. Graff. UCI machine learning repository, 2017.

[10] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[11] C. Garbin and O. Marques. Assessing methods and tools to improve reporting, increase transparency, and reduce failures in machine learning applications in health care. *Radiology: Artificial Intelligence*, 4(2):e210127, 2022.

[12] Bryce Goodman. A Step Towards Accountable Algorithms?: Algorithmic Discrimination and the European Union General Data Protection. *29th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.*, (Nips):1–7, 2016.

[13] S. Guiasu and A. Shenitzer. The principle of maximum entropy. *The mathematical intelligencer*, 7(1):42–48, 1985.

[14] Yang Guo, Shuhui Liu, Zhanhuai Li, and Xuequn Shang. Bcdforest: a boosting cascade deep forest model towards the classification of cancer subtypes based on gene expression data. *BMC bioinformatics*, 19(5):1–13, 2018.

[15] K. Hajian-Tilaki. Receiver operating characteristic (roc) curve analysis for medical diagnostic test evaluation. *Caspian journal of internal medicine*, 4(2):627, 2013.

[16] A. Halevy, P. Norvig, and F. Pereira. The Unreasonable Effectiveness of Data. *IEEE Intelligent Systems*, 24(2):8–12, 2009.

[17] T. H. Haveliwala. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE Trans. Knowl. Data Eng.*, 15(4):784–796, 2003.

[18] E. B. Hekler, P. Klasnja, G. Chevance, N. M. Golaszewski, D. Lewis, and I. Sim. Why we need a small data paradigm. *BMC Medicine*, 17(1):133, 2019.

[19] M. K. Ho, D. Abel, T. L. Griffiths, and M. L. Littman. The value of abstraction. *Current Opinion in Behavioral Sciences*, 29:111–116, 2019.

[20] M. Hsu, Y. I. Chang, and H. Hsueh. Biomarker selection for medical diagnosis using the partial area under the roc curve. *BMC research notes*, 7(1):1–15, 2014.

[21] W. Jin, T. Derr, H. Liu, Y. Wang, S. Wang, Z. Liu, and J. Tang. Self-supervised Learning on Graphs: Deep Insights and New Direction. *arXiv*, 2020.

[22] Mark MacCarthy. An Examination of the Algorithmic Accountability Act of 2019. *SSRN Electronic Journal*, pages 1–10, 2020.

[23] T. Maszczyk and W. Duch. Comparison of shannon, renyi and tsallis entropy used in decision trees. In *Artificial Intelligence and Soft Computing - ICAISC 2008, 9th International Conference, Zakopane, Poland, June 22-26, 2008, Proceedings*, volume 5097 of *Lecture Notes in Computer Science*, pages 643–651. Springer, 2008.

[24] C. E. Metz. Basic principles of roc analysis. *Seminars in Nuclear Medicine*, 8(4):283–298, 1978.

[25] M. Mitchell. Why AI is Harder Than We Think. *arXiv*, 2021.

[26] M. Mitchell and D. C. Krakauer. The Debate Over Understanding in AI's Large Language Models. *arXiv*, 2022.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[28] T. P. Peixoto. Nonparametric bayesian inference of the microcanonical stochastic block model. *Physical Review E*, 95(1):012317, 2017.

[29] Gabriëlle Ras, Marcel van Gerven, and Pim Haselager. Explanation Methods in Deep Learning: Users, Values, Concerns and Challenges. *Explainable and interpretable models in computer vision and machine learning*, pages 19–36, 2018.

[30] J. Repplinger. G.G. chowdhury. *Introduction to Modern Information Retrieval.* 3rd ed. london: Facet, 2010. 508p. alk. paper, $90 (ISBN 9781555707156). LC2010-013746. *Coll. Res. Libr.*, 72(2):194–195, 2011.

[31] G. S. Sampaio, A. R. A. V. Filho, L. S. da Silva, and L. A. da Silva. Prediction of motor failure time using an artificial neural network. *Sensors*, 19(19):4342, October 2019.

[32] C. M. Sauer, L. Chen, S. L. Hyland, A. Girbes, P. Elbers, and L. A. Celi. Leveraging electronic health records for data science: common pitfalls and how to avoid them. *The Lancet Digital Health*, 4(12):e893–e898, 2022.

[33] S. M. Shah and R. A. Khan. Secondary Use of Electronic Health Record: Opportunities and Challenges. *IEEE Access*, 8:136947–136965, 2020.

[34] R. Shao and X. Bi. Transformers meet small datasets. *IEEE Access*, 10:118454–118464, 2022.

[35] R. Shwartz-Ziv, R. Balestriero, and Y. LeCun. What Do We Maximize in Self-Supervised Learning? *arXiv*, 2022.

[36] V. Sigillito, S. Wing, L. Hutton, and K. Baker. Ionosphere. UCI Machine Learning Repository, 1989.

[37] Eduardo Soares and Plamen Angelov. Fair-by-design explainable models for prediction of recidivism. pages 3–7, 2019.

[38] D. Spathis, I. Perez-Pozuelo, L. Marques-Fernandez, and C. Mascolo. Breaking away from labels: The promise of self-supervised machine learning in intelligent health. *Patterns*, 3(2):100410, 2022.

[39] W. Wolberg, W. Street, and O. Mangasarian. Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository, 1995.

[40] W. Xu, G. Yu, A. Zare, B. Zurweller, D. Rowland, J. Reyes-Cabrera, F. B. Fritschi, R. Matamala, and T. E. Juenger. Overcoming small minirhizotron datasets using transfer learning. *Comput. Electron. Agric.*, 175:105466, 2020.

[41] T. Zhou, H. Dou, J. Tan, Y. Song, F. Wang, and J. Wang. Small dataset solves big problem: An outlier-insensitive binary classifier for inhibitory potency prediction. *Knowl. Based Syst.*, 251:109242, 2022.

[42] Zhi-Hua Zhou and Ji Feng. Deep forest: Towards an alternative to deep neural networks. In *IJCAI*, pages 3553–3559, 2017.

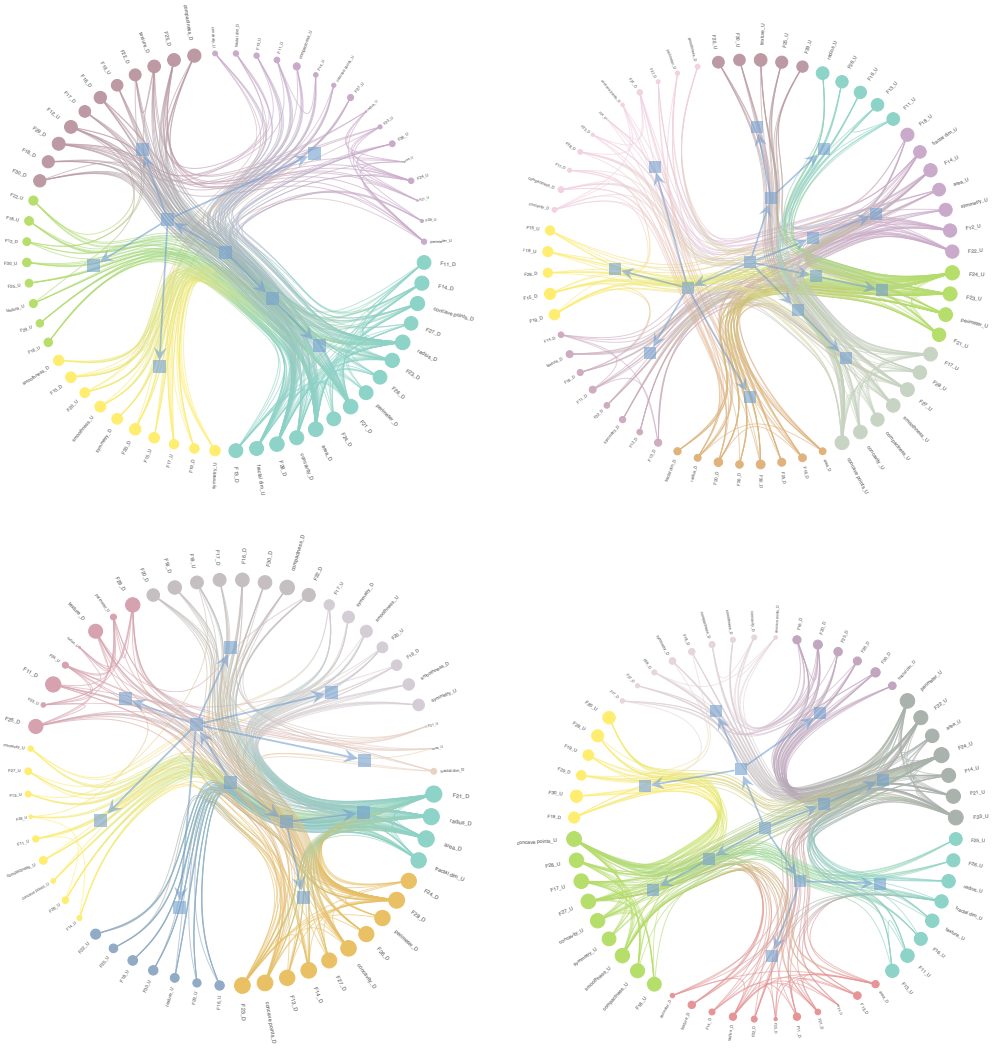

Figure 2: Examples of SMs (top) and SAMs (bottom) stratified by class 0 (left) and class 1 (right)

Missing Values	Accuracy	Balanced Accuracy
Ionosphere		
0%	0.9173789173789174	0.9111111111111111
1%	0.8888888888888888	0.8801587301587301
5%	**0.905982905982906**	**0.8987301587301587**
10%	**0.9088319088319088**	**0.900952380952381**
50%	**0.8831908831908832**	**0.8722222222222222**
Sonar		
0%	**0.8509615384615384**	**0.8473576669452958**
1%	**0.8509615384615384**	**0.8473576669452958**
5%	**0.8557692307692307**	**0.8525123061205535**
10%	**0.8413461538461539**	**0.8363982539240271**
50%	**0.8846153846153846**	**0.8840902758428532**
Wisconsin Breast Cancer		
0%	0.9525483304042179	0.9526055705300989
1%	**0.9525483304042179**	**0.9526055705300989**
5%	**0.9525483304042179**	**0.9506897098462026**
10%	**0.9578207381370826**	**0.958723111886264**
50%	**0.9525483304042179**	**0.9516476401881507**
Accelerometer		
0%	0.6763823529411764	0.6763823529411765
1%	0.6755	0.6755
5%	0.671421568627451	0.671421568627451
10%	0.6655686274509804	0.6655686274509804
50%	0.6218921568627451	0.6218921568627451
HIGGS		
0%	0.62736	0.6238901691324948
1%	0.62751	0.624098083884083
5%	0.62357	0.6205104437510285
10%	0.62275	0.620051722051421
50%	**0.59962**	**0.5975619384689126**
Air Pressure System Failure		
0%	0.92815	**0.9157881355932204**
1%	0.9280166666666667	**0.9152288135593221**
5%	0.9280166666666667	**0.9167033898305085**
10%	0.92905	**0.9157542372881355**
50%	0.9289333333333334	**0.9186440677966101**

Table 2: SaNDA results.

Missing Values	Accuracy	Balanced Accuracy
Ionosphere		
0%	**0.9528301886792453**	**0.9358974358974359**
1%	**0.9150943396226415**	**0.895331037122082**
5%	0.9056603773584906	0.8878683505549176
10%	0.9056603773584906	0.8932261768082663
50%	0.8207547169811321	0.7885572139303483
Sonar		
0%	0.8412698412698413	0.8321428571428571
1%	0.7619047619047619	0.7571428571428571
5%	0.7936507936507936	0.7928571428571429
10%	0.7301587301587301	0.7392857142857143
50%	0.6984126984126984	0.6928571428571428
Wisconsin Breast Cancer		
0%	**0.9590643274853801**	**0.9576719576719577**
1%	0.9415204678362573	0.9338624338624338
5%	0.9473684210526315	0.9484126984126984
10%	0.9473684210526315	0.9451058201058201
50%	0.9181286549707602	0.9054232804232805
Accelerometer		
0%	**0.8890522875816993**	**0.8890515915396984**
1%	**0.8828431372549019**	**0.882842709040509**
5%	**0.86359477124183**	**0.8635939054043376**
10%	**0.8401633986928104**	**0.840161316877716**
50%	**0.7127450980392157**	**0.7127476992807777**
HIGGS		
0%	**0.6681**	**0.6663068303679973**
1%	**0.6641**	**0.6619667971684573**
5%	**0.6508666666666667**	**0.6486250092041421**
10%	**0.6446666666666667**	**0.6426414972197971**
50%	0.5959666666666666	0.5935445254284073
Air Pressure System Failure		
0%	**0.9921666666666666**	0.8202597442446821
1%	**0.9915**	0.8101564587310722
5%	**0.9890555555555556**	0.7552099275631847
10%	**0.9884444444444445**	0.7337432523149621
50%	**0.9855**	0.6785424551283155

Table 3: Random Forest results.

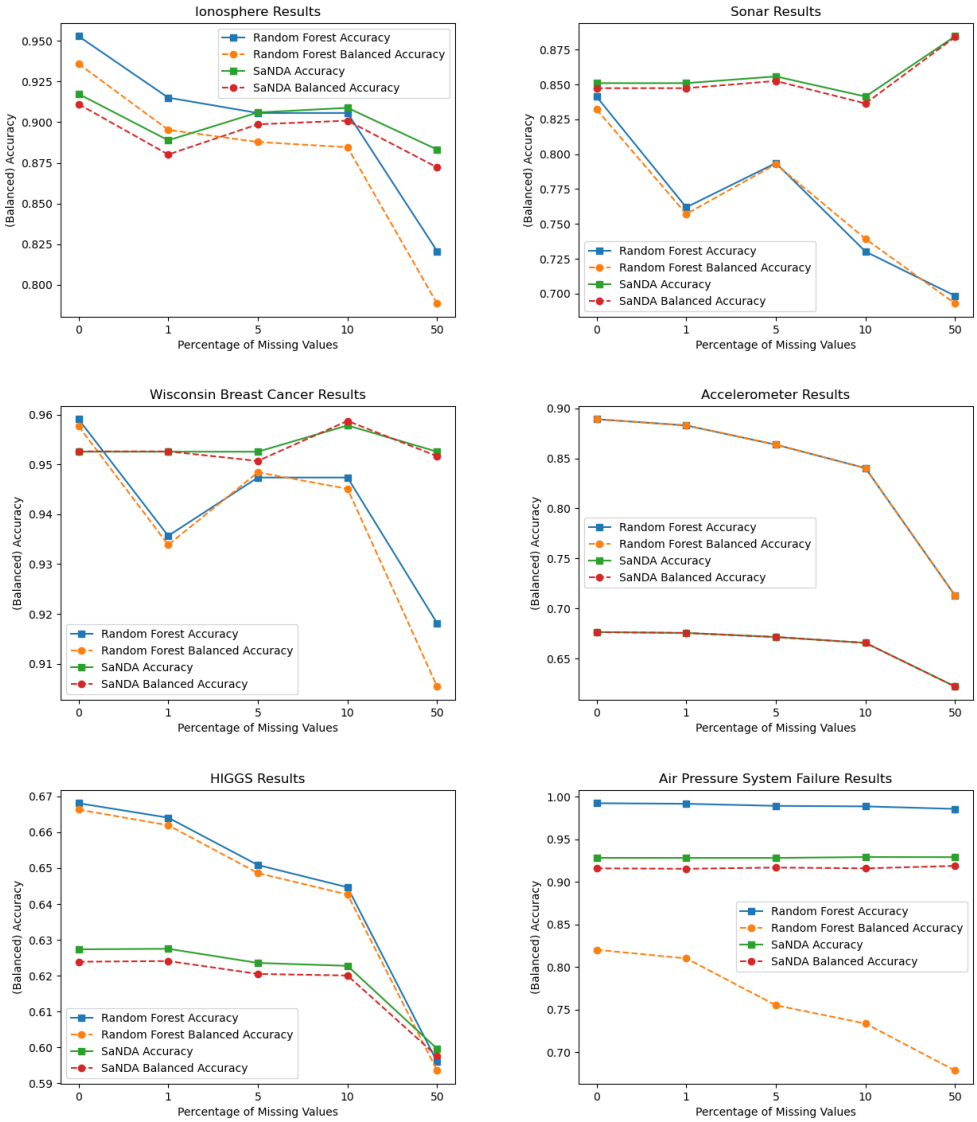

Figure 3: Experiment results per dataset vs Random Forest.

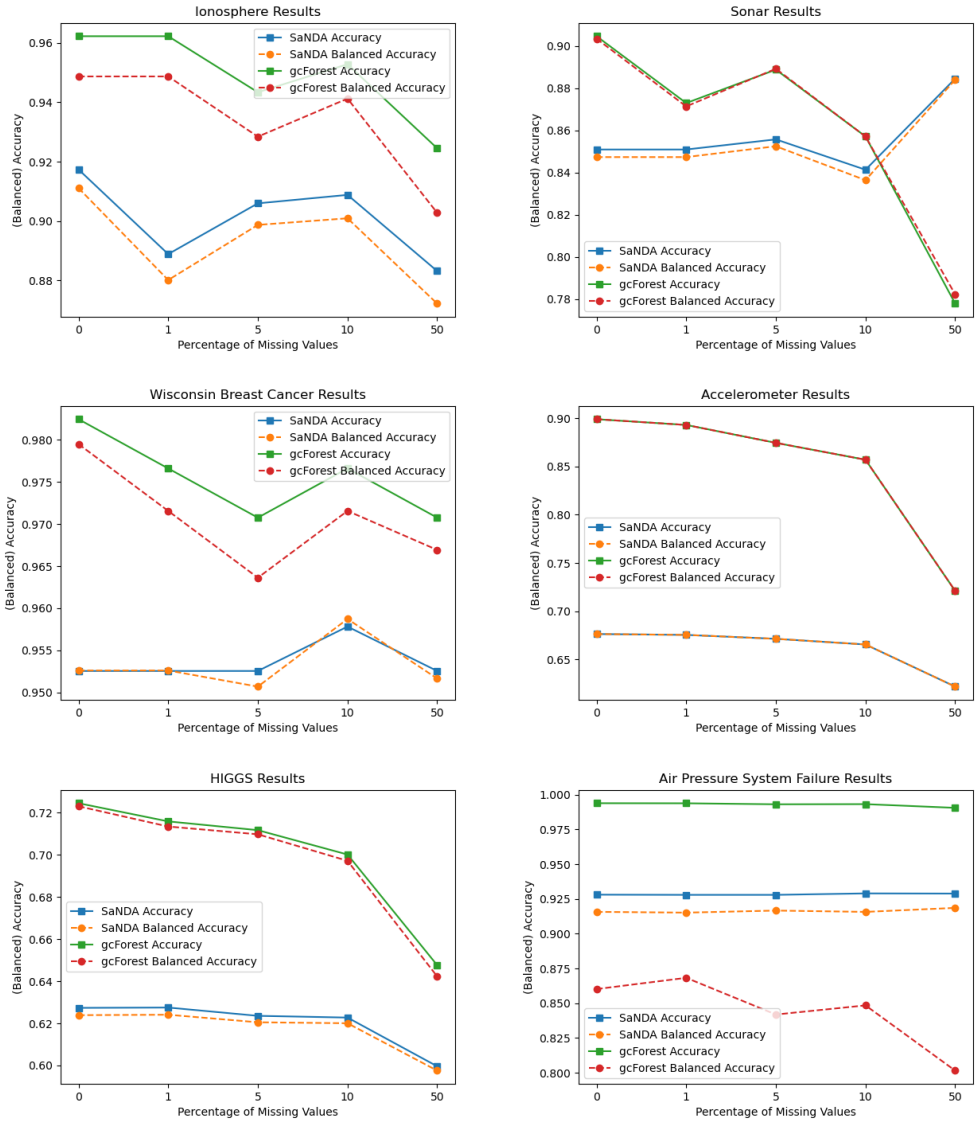

Figure 4: Experiment results per dataset vs gcForest.

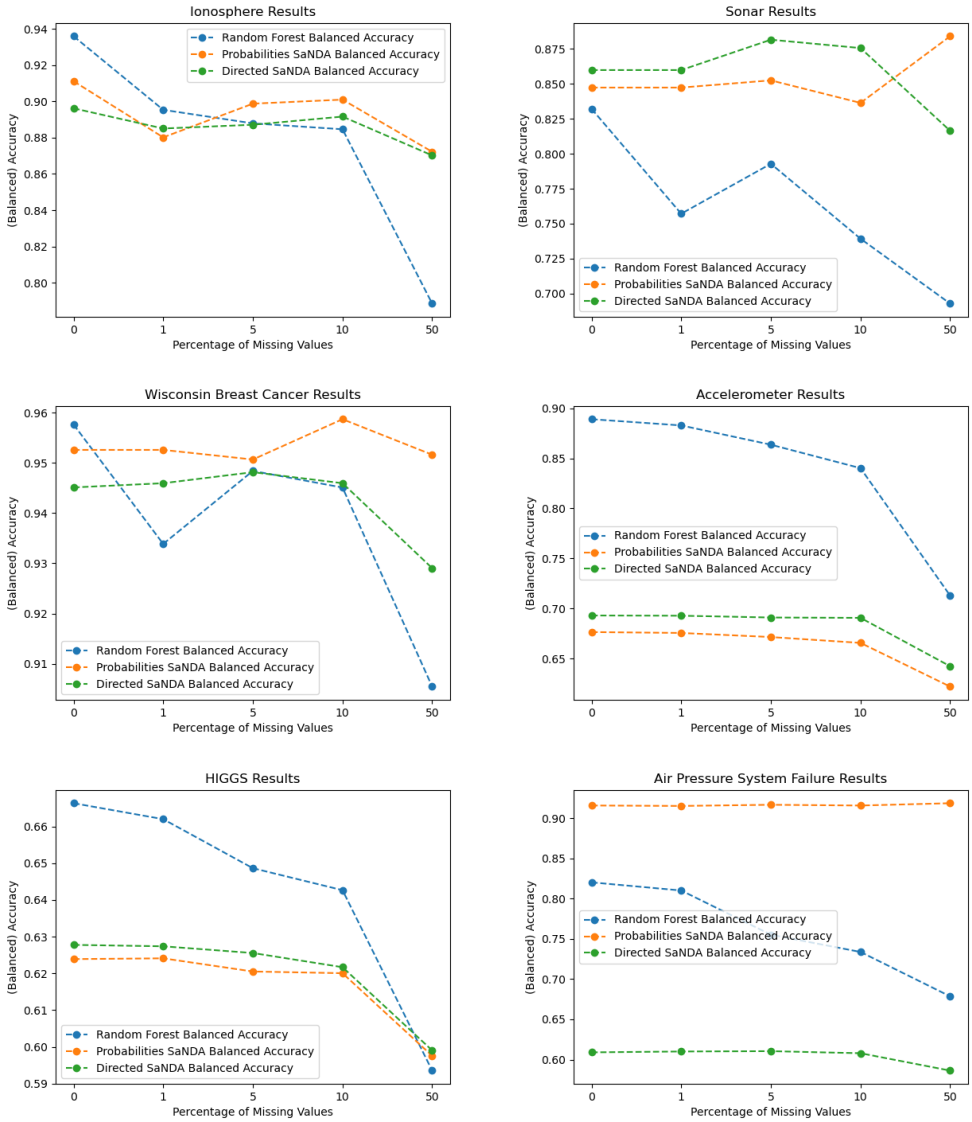

Figure 5: Comparison between using probabilities and directed PageRank.