DE GRUYTER OPEN

JAISCR, 2018, Vol. 8, No. 1, pp. 31 – 48
10.1515/jaiscr-2018-0002

# CLASSIFIERS ACCURACY IMPROVEMENT BASED ON MISSING DATA IMPUTATION

Ivan Jordanov, Nedyalko Petrov, Alessio Petrozziello

*School of Computing, University of Portsmouth*
*Portsmouth, PO1 3FE, UK*
*ivan.jordanov@port.ac.uk, nedyalko.petrov@port.ac.uk,*
*alessio.petrozziello@port.ac.uk*

*Submitted: 14th February 2017; Accepted: 28th March 2017*

**Abstract**

In this paper we investigate further and extend our previous work on radar signal identification and classification based on a data set which comprises continuous, discrete and categorical data that represent radar pulse train characteristics such as signal frequencies, pulse repetition, type of modulation, intervals, scan period, scanning type, etc. As the most of the real world datasets, it also contains high percentage of missing values and to deal with this problem we investigate three imputation techniques: Multiple Imputation (MI); K-Nearest Neighbour Imputation (KNNI); and Bagged Tree Imputation (BTI). We apply these methods to data samples with up to 60% missingness, this way doubling the number of instances with complete values in the resulting dataset. The imputation models performance is assessed with *Wilcoxon*'s test for statistical significance and *Cohen*'s effect size metrics. To solve the classification task, we employ three intelligent approaches: Neural Networks (NN); Support Vector Machines (SVM); and Random Forests (RF). Subsequently, we critically analyse which imputation method influences most the classifiers' performance, using a multiclass classification accuracy metric, based on the area under the ROC curves. We consider two superclasses ('military' and 'civil'), each containing several 'subclasses', and introduce and propose two new metrics: inner class accuracy (IA); and outer class accuracy (OA), in addition to the overall classification accuracy (OCA) metric. We conclude that they can be used as complementary to the OCA when choosing the best classifier for the problem at hand.

**Keywords:** machine learning, missing data, model-based imputation, neural networks, random forests, support vector machines, radar signal classification.

## 1 Introduction

Missing values are unavoidable in real world datasets and the radar once make no exception. There is a variety of causes why data may be missing, but common ones are related to human errors, equipment faults or to coarse environment conditions that result in noise and propagation distortion, leading to incomplete, incorrect or missed intercepted signals. Dealing with it is important part of the data pre-processing phase in any machine learning task and can lead to improved accuracy in the later stages of classification [1].

Mechanisms of missing data usually belong to three categories [1]: missing at random (MAR) – where the missingness can be fully accounted for by the variables where the information is complete; missing completely at random (MCAR), where the missingness is unrelated both to its value or to the value of any other variables (observed and un-

observed); and missing not at random (MNAR) - where the missingness depends only on unobserved data. The last group (MNAR), typically yields biased parameter estimates, while MCAR and MAR analyses lead to unbiased ones (however, MCAR still has a loss of statistical power).

Dealing with missingness requires an analysis strategy leading to least biased estimates, without losing statistical power of the dataset. However, the nature of those two criteria is contradictory and using the information from the partially completed data (keeping their statistical power), while imputing the missing values, inevitably brings biases.

There are two groups of methods widely used to solve the missing data problem: Deletion methods and Model-based methods [1-3]. The first group includes pairwise and listwise deletion. The pairwise deletion (also called "unwise" deletion) keeps as many cases as possible for each analysis (this way making use of all available information), but the problem is that the analyses are incomparable because each of them is based on different subsets of data, with different sample sizes and different standard errors. The listwise deletion (or complete case analysis) is a simple approach which discards all records with missing data. The advantages of this technique include comparability across the analyses and the fact that it does not introduce biased parameter estimates (assuming the data is MCAR), while the disadvantage is that there may be substantial loss of statistical power (especially if there are many excluded samples).

In this work we consider three model-based imputation approaches: Bagged Tree Imputation (BTI), K-Nearest Neighbour Imputation (KNNI) and Multiple Imputation (MI).

The BTI with gradient boosting [4] is a machine learning technique for solving regression problems, which produces a prediction model ensembling weak ones. For each predictor that has missing data, a tree model is trained, based on the other predictors and the values are imputed using a regression function.

The KNNI [3] approach comprises three phases: the first one is to take only the subset without missing data and to use it as a prototype dataset to select the nearest one; the second phase is to choose a distance/similarity metric and compute it between each pattern with missing data and the full dataset; and finally, the third one is to impute the data, using the mean, the median or the mode among the chosen neighbours.

The MI approach involves three distinct stages: firstly, sets of plausible data for the missing observations are created and used to construct many completed datasets (without missing values); secondly, each of these datasets is independently considered using standard statistical analysis for complete datasets; and thirdly, all the sets from the previous stage are combined into one estimate for the inference. The aim of the MI process is not just to fill in the missing values with plausible estimates, but also to preserve important characteristics of the whole dataset. As with most (multiple regression) prediction models, the danger of overfitting is real, usually leading to less generalizable results compared to the original data [2].

The remainder of the paper is organized as follows: Section 2 provides additional background on radar signal classification and discusses prior work in this area; Section 3 introduces briefly the used classification methods; Section 4 gives the problem statement and data set analysis; Section 5 presents the employed data imputation models and the data coding and transformation; Section 6 illustrates the data imputation with two experiments – label imputation and continuous feature imputation; classifiers training and obtained results are reported and critically discussed in Section 7 (we also introduce two new metrics: inner (super) class error and outer class error to assess the classifiers accuracy); and finally, Section 8 concludes the discussion and gives potential avenues for further research.

## 2 Radar Signal Classification

Over the years, radars have been used in various areas of application, but the two broad ones are military and civil. In the military sector, radar activity has found application in surveillance, navigation and weapon guidance (detecting, locating, tracing, and identifying air, marine, and terrestrial objects, at small to medium and large distances). In the civil area, radars are widely used for traffic control, navigation, weather forecast, pollution monitoring, space observation, sport systems, and others [5-7].

The radar characteristics (range, resolution and sensitivity) are determined by its transmitter and waveform generator. Most radars operate within the micro-wave region of the electromagnetic spectrum and are used for short range applications with high resolution, while other radars operate in the radio-wave region and are usually preferred for longer range purposes [5].

Radar detection, classification and tracking of targets against a cluttered background are considered as a 'general radar problem'. For military purposes 'the general radar problem' involves the interception, localisation, analysis and identification of radiated electromagnetic energy - also known as radar Electronic Support Measures (ESM). They are considered a reliable source of valuable information for threat detection, threat avoidance, and in general, situation awareness for timely deployment of counter-measures [8].

A real-time identification of a radar emitter associated with each intercepted pulse train is a very important function of the radar ESM. Typical approaches include sorting radar pulses into individual pulse trains, then comparing their features with a library of parametric descriptions to get a list of likely radar types. This can be a very difficult task, as there may be radar modes with no records in ESM library; increases in environment density (e.g., Doppler spectrum radars transmitting hundreds of thousands of pulses per second); overlaps of different radar type parameters; agility of radar features, such as radio frequency and scan, pulse repetition interval, etc.; multiplication and dispersion of the modes for military radars; noise and propagation distortion that lead to incomplete or incorrect signals [9].

Intercepted and collected pulse train properties typically include signal frequencies, type of modulation, pulse repetition intervals, etc. The collected information usually consists of a combination of continuous, discrete and categorical data, and frequently includes missing entries. Table 1 shows several samples of radar data that illustrates the above mentioned characteristics and includes missing values as well. Handling the missing data is very important part of the pre-processing stage of any machine learning experiment, before applying the considered classification models [10].

# 3 Intelligent Techniques in Radar Identification

## 3.1 Neural Networks

A number of approaches have been investigated and applied to solve the radar emitter recognition and identification problem and substantial part of this research includes Neural Networks (NN), because of their flexibility, fault tolerance and capability to handle incomplete and noisy radar data. NN models have previously been applied to solve different tasks of radar ESM processing and more recently, many new radar and target recognition systems comprise neural networks as a key classifier [11]. Previous works using a variety of NN architectures and topologies for radar identification, recognition and classification based on ESM data include popular Multilayer Perceptron (MLP), Radial Basis Function (RBF) based NN, Support Vector Machines (SVM), single parameter dynamic search NN, [11-13], and deep learning NN [14].

For example, in [9] the authors use initial clustering algorithm to separate pulses from different emitters according to position-specific parameters of the input pulse stream when implementing their "What-and-Where fusion strategy" and then apply fuzzy ARTMAP neural network to classify streams of pulses according to radar type, using their functional parameters. They also do simulations with dataset having missing input pattern components and missing training classes, incorporating a bank of Kalman filters to demonstrate high level performance of their system on incomplete, overlapping and complex radar data. The work presented in [11] investigates the potential of NN (MLPs) when used in Forward Scattering Radar (FSR) applications for target classification. The authors analyse radar signal data and extract features, which are then used to train NN for target classification. They also apply the more commonly used K-Nearest-Neighbour classifier to compare the results from the two approaches and conclude that the NN solution has better accuracy. In [13] the authors investigate the potential of NN (MLP) when used in Forward Scattering Radar (FSR) applications for target classification. In [14] deep NN architectures are employed for SAR images recognition, while in [15] a vector neural network is applied for emitter identification. In many cases the NN are hybridized with

**Table 1**. Sample radar data subset. Missing values (i.e., values that could not have been intercepted or recognized) are denoted with '-'. The rest of the acronyms are defined in Table 2.

| ID | FN | RFC | RFmi | RFmi | PRC | PRImi | PRIma | PDC | PDmi | PDma | ST | SPmi | SPma |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 84 | SS | B | 5300 | 5800 | K | – | – | S | – | – | A | 5.9 | 6.1 |
| 4354 | AT | F | 2700 | 2900 | F | 1351.3 | 1428.6 | S | – | – | A | 9.5 | 10.5 |
| 7488 | 3D | B | 8800 | 9300 | K | 100 | 125 | S | 13 | 21 | B | 1.4 | 1.6 |
| 9632 | WT | F | 137 | 139 | T | – | – | V | – | – | D | – | – |
| 9839 | 3D | S | 2900 | 3100 | J | – | – | V | 99 | 101 | A | 9.5 | 10.5 |

**Table 2**. Data description and percentage of missing values. In column 'Type': I – integer; C – categorical; R – real values

| Field | Field Description | Type | Levels | % Missing |
|---|---|---|---|---|
| **ID** | Reference for the line of data | I | - | - |
| **FN** | Function performed by the radar ('*3D*' – 3D surveillance, '*AT*' – air traffic control, '*SS*' – surface search, '*WT*' – weather tracker, etc.) | C | 142 | 1.4 |
| **RFC** | Type of modulation used by the radar to change the frequency from pulse to pulse ('*A*' – agile, '*F*' – fixed, etc.) | C | 12 | 20.7 |
| **RFmi** | Min frequency used by the radar | R | - | 11.2 |
| **RFma** | Max frequency used by the radar | R | - | 11.2 |
| **PRC** | Type of modulation used by the radar to change the Pulse Repetition Interval (PRI), ('*F*' – fixed, etc.) | C | 15 | 15 |
| **PRImi** | Min PRI used by the radar | R | - | 46.7 |
| **PRIma** | Max PRI used by the radar | R | - | 46.7 |
| **PDC** | Type of modulation used by the radar to change the pulse duration ('*S*' - stable) | C | 5 | 12.9 |
| **PDmi** | Min pulse duration used by the radar | R | - | 46.1 |
| **PDma** | Max pulse duration used by the radar | R | - | 46.1 |
| **ST** | Scanning type – used method by the radar to move the antenna beam ('*A*' – circular, '*B*' – bidirectional, '*W*' – electronically scanned, etc.) | C | 28 | 11.3 |
| **SPmi** | Min scan period used by the radar | R | - | 59.4 |
| **SPma** | Max scan period used by the radar | R | - | 59.4 |

other techniques such as fuzzy systems, clustering algorithms, wavelet packets, Kalman filters, particle swarm optimization-based SVM, etc., which in turn leads to recognition systems with increased accuracy [9,15].

### 3.2 Support Vector Machines

Support Vector Machines (SVM) are learning machines based on the statistical learning theory, that can use linear, polynomial, Gaussian, exponential and hybrid kernels for the classification, Radial Basis Function (RBF) networks, and even particle swarm optimization in some cases [16]. They minimise the structure risk in a higher dimensional feature space, searching for the hyperplane with the largest margin between the classes. In other words, if the feature space is non-linearly separable, the SVM use non-linear mapping to find an optimal classification hyperplane. The choice of a map function (kernel) can significantly influence the classification results [17]. One advantage of this approach is that it is possible to design and use a kernel function for a specific task that could be applied to the data without the need of a previous feature extraction process.

The SVM have been used recently to classify radar pulse signals. For example, in [18] the authors use SVM to classify targets by analysing micromotions of an object having single signature. Target recognition method based on SVM with hybrid differential evolution and self-adaptive particle swarm optimization is investigated in [16]. The reported low error rate results in these works showed that the SVM are useful approach for solving radar classification and recognition problems.

In this work we used the SVM implementation from the R package 'E1071', which we found to work faster on multi-class problems [19].

### 3.3 Random Forests

Random Forests (RF) [20] is a machine learning technique that builds a multitude of weak decisional trees at training time and outputs the class that is the mode of the classes (classification) or average prediction (regression) of the individual trees. Each tree is individually trained on a sample of the training data, and at each node, the algorithm only searches across a random subset of the features to determine a split. The input vector to be classified is submitted to each of the decision trees in the forest and the prediction is then formed using a majority vote. The method combines *Breiman*'s "bagging" idea [20], and the random selection of features, introduced independently by [21].

Some authors [22] consider random forests as "clearly the best family of classifiers", while others [23] argue that such statement is flawed and question their conclusion, showing that the study's own statistical tests indicate that "RF do not have significantly higher percent accuracy than support vector machines and neural networks". Nevertheless, most authors agree that key advantages of RF comprise: the capability of determining variable importance; avoiding decision trees' habit of overfitting; and a high classification accuracy [20]. However, as the split rules for classification are unknown, the RF could be considered as a black box type classifier. In our implementation, we used the R 'randomForest' package, which implements the *Breiman*'s random forests algorithm [20].

## 4 Problem Statement and Data Set Analysis

Accurate and real-time classification of radar signals is of crucial importance for timely threat detection, threat avoidance, and deployment of countermeasures. In this context, this paper investigates the potential application of three methods for identification of radar types, associated with intercepted pulse trains.

In our previous work [24], we initially used a deletion technique (listwise) to obtain 7693 complete samples from a total of about 30000 intercepted generic data samples and subsequently adopted multiple imputation (MI), to take advantage of the whole dataset when solving identification and classification tasks. Each signals in the dataset is pre-classified by experts into one of 125 categories, based on the main radar emitter functions (e.g., weather tracking, surveillance, air traffic control, air defence, etc.).

As mentioned above, in this work we investigate three different state-of-the-art classifiers: NN; SVM; and RF; as well as three imputation techniques: MI; KNNI; and BTI. Firstly, we employ MI

using 15656 samples, including the recovered 7963 samples with missing data (doubling the number of the complete available data). When employing KNNI and BTI we managed to recover all missing data and used 29094 samples (the whole dataset). From the missing data samples (example given in Table 1), we excluded only those with above 60% missingness and used MI, RNNI, and BTI for substituting the missing data.

Table 1 contains all the relevant information of the dataset at hand. The first two columns include the data sample identifier and the category label (specifying the radar function and considered as the output to classify), followed by the radar signal pulse train characteristics (e.g., type of modulation, signal frequencies, pulse repetition intervals, etc.), which are considered as input features.

A summary of the data distribution is presented in Table 2, where an overview of the type, range and percentage of missing values in the dataset is also given. The included data consists of both numerical (discrete and continuous) and categorical values, the latter of which we coded during the data pre-processing stage, using different representations (i.e., continuous, binary and dummy). The table also shows that the percentage of missingness varies from 11.2% to 59.4% (for the radar frequency and scan period features respectively).

# 5 Data Imputation and Pre-processing

The pre-processing of the available data is of primarily importance in machine learning as it can significantly affect the overall accuracy of the employed classification algorithm. The main objective is to analyse inconsistences and outliers in the data, and to facilitate the underlying mathematical apparatus of the machine learning method, leading to an overall improvement of the classifier's performance in terms of faster convergence and higher accuracy.

## 5.1 Bagged Tree Imputation (BTI)

Bagging predictors approach generates manifold versions of a predictor to get an aggregated one. The aggregation function usually is the average value over all predictor estimations when for a numerical outcome, or employs a majority vote when the desired output is a categorical one. The multiple predictions are estimated by bootstrapping from the training set and subsequently using these as new learning sets. Tests on real and artificial data sets, using classification and regression trees and subset bootstrap with linear regression, show that bagging can benefit the accuracy. Vital component of this technique is the instability of the prediction model, but if perturbing the learning set can cause significant changes in the constructed predictor, then bagging can improve the accuracy [10], [25].

BTI is robust to outliers and is able to impute the data very accurately using surrogate splits, which are essential in handling the missing data [26]. For example, if a decision tree is trained to predict variable $p$, using a set of features $a$, $b$ and $c$, and if for a new data point there are values only for $a$ and $b$, the missing value of $c$ would raise problems for the prediction of $p$. In such cases, the models are trained to include surrogate splits. So, whenever the variable $c$ is missing, the algorithm defers the decision to another variable that is highly correlated to $c$, which will allow the prediction to continue. Another important feature of the tree model is its flexibility, with the RF we can train different models and postpone the real prediction to a system vote among the models.

In this work we employ gradient boosting technique for the values regression, which is based on an ensemble of weak decision trees, implementing the R "gbm" package [4] for the imputation function.

**Algorithm 1**. Pseudocode of BTI

```
Input: X (data matrix)
Repeat:
In parallel:
For each predictor p in X, train a tree model
    that predicts p using all other p-1 features
Predict missing values for predictor p, but do
    not write to X yet
Sync threads
Fill X with predicted values
```

## 5.2 K-Nearest Neighbour Imputation (KNNI)

Another imputation approach used in this work is the K-Nearest Neighbour (KNNI) [3], [27]. This technique has several benefits: the method can pre-

dict both continuous (the average among the nearest neighbours) and categorical variables (the most frequent value); there is also no need to build a model beforehand (as in the BTI).

If the available data set is denoted with X and the complete subset with $X_c$, for each sample containing a missing point, the procedure will search the K most similar records in the $X_c$ dataset, according to the adopted distance measure, before imputing the missing value. This means that there is a need to separate the complete dataset from the initial one containing missing data, define an appropriate distance measure, and for each sample in the missing dataset to look for the nearest neighbours, using that measure. Then, the mode value is used when substituting a categorical variable and the average value for a continuous one. In this work, we use the function kNN of the R package VIM [28].

### 5.3 Multiple Imputation

For imputing the missing multivariate data, we use sequential multiple imputation algorithm [29], implemented in the *impSeq* function from the ***R*** package (we also tried two other R functions: *impNorm;* and *impSeqRob*; but they did not produce better results when tested on the complete dataset).

If $X_c$ is a subset with no missing data, derived from the available data set X, the procedure will start with $X_c$ to estimate sequentially the missing values of an incomplete observation x*, by minimizing the covariance of the augmented data matrix X* = ($X_c$, x*). Subsequently, the data sample x* is added to the complete data subset and the algorithm continues with the estimate of the next data sample with missing values.

**Algorithm 2.** Pseudocode of MI

1: Input: X (data matrix), $X_c$ (complete subset of X)
2: For each x* (incomplete observation in X):
3: x = Min var( ($X_c$, x* )
4: Add x to $X_c$

Because *impSeq* uses the sample mean and covariance matrix, it is vulnerable to the presence of outliers, but this can be enhanced by including robust estimators of location and scatter (which is realised in *impSeqRob* function) [29]. Because the outlyingness metric can be computed for a complete dataset only, firstly the sequential imputation of the missing data is done and then the outlyingness measure is computed and used to define whether the observation is an outlier or not. If the measure does not exceed a predefined threshold, the observation is included in the next steps of the algorithm (nevertheless, the use of *impSeqRob* in our case did not produce better results when tested on complete dataset, which may be simply because of the lack of outliers).

As we mentioned before, in the available radar dataset of about 30000 samples, there are 7693 fully intercepted and recognised radar signals that constitutes the complete subset (received after listwise deletion of the original dataset) [24]. Subsequently, employing the MI on the missing data samples with less than 60% missingness, led to dataset of 15656 observations, which exceeds the doubled size of the initial data subset. Processing the subset with missingness with the KNNI and the BTI tripled the complete data samples, enabling us to utilise valuable information and use the statistical power of the data contained in the samples with missing values.

The applied supervised learning for the identification and classification of the radar signals uses from two to eleven output classes: two classes for the first set of simulations - civil and military; and eleven (four civil and seven military classes) in the second set of simulations (defined by experts in the field from a total of 125 functional categories). Table 3 shows the samples from Table 1 with the inputted values produced by the implemented MI.

### 5.4 Data Coding and Transformation

This stage of the pre-processing aims to transform the data into a suitable form for feeding the chosen classifier. A transformation coding is applied to convert the categorical features to numerical ones. Three of the most broadly applied coding techniques are implemented: continuous; binary; and dummy. In the first case, each of the categorical values is substituted with an ordinal number, e.g., the 12 categories for the RFC input are encoded within (1-12) interval, the 15 PRC categories within (1-15) interval, etc.

A sample of a data subset coded with continuous values is given in Table 4. With the Binary coding, each categorical value is substituted by $\log_2$N

**Table 3**. Sample radar data subset with imputed values for the missing continuous values

| ID | FN | RFC | RFmi | RFma | PRC | PRImi | PRIma | PDC | PDmi | PDma | ST | SPmi | SPma |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 84 | SS | B | 5300 | 5800 | K | 963.2 | 5625 | S | 5.8 | 17 | A | 5.9 | 6.1 |
| 4354 | AT | F | 2700 | 2900 | F | 1351 | 1428 | S | 4 | 6.3 | A | 9.5 | 10.5 |
| 7488 | 3D | B | 8800 | 9300 | K | 100 | 125 | S | 13 | 21 | B | 1.4 | 1.6 |
| 9632 | WT | F | 137 | 139 | T | 622.6 | 31312 | V | 61.1 | 93.1 | D | 12 | 47.8 |
| 9839 | 3D | S | 2900 | 3100 | J | 2058 | 48128 | V | 99 | 101 | A | 9.5 | 10.5 |

new binary variables (where N is the number of categories taken by that variable), each having value of either 0 or 1 (illustrated in Table 5 for 32 categories).

Finally, the non-numerical attributes are coded using dummy variables. In particular, every *p* levels of a categorical variable are represented by introducing *p* dummy variables. An example of dummy coding for 32 categorical levels is shown in Table 6.

To balance the impact of the different input parameters on the training algorithm, the data was scaled during the pre-processing phase. Correspondingly, each of the conducted experiments in the next section is evaluated using three forms of the input data set: the original data (with no scaling); normalized data (i.e., within (0, 1) interval); and standardized data (i.e., 0-mean and 1-variance).

## 6 Assessment Baseline

To begin with, it is essential to determine which imputation method leads to the best substitution of the missing values. For this purpose, we designed two experiments, as described below.

### 6.1 Label Imputation

To test the reliability of the investigated imputation models, 25% of the labels in the complete subset (˜4000 records), were randomly removed. The labels were subsequently imputed, using the three methods: BTI, KNNI and MI. To cope with the random nature of the algorithms, we run each imputation 30 times. This was done for the two simulations: with 2 classes and with 11 classes.

In the 2 class imputation case, the KNNI method showed the best performance with 90% accuracy, followed by BTI with 84% and MI with 65% (Figure 1). We think that KNNI achieved the best accuracy due to the skewed nature of the data: in fact, 75% of the data belongs to the second class, this way, the unbalanced dataset helps the search of nearest neighbours to go towards the class with more samples. On the contrary, the MI and BTI methods assume balanced label distribution, which probably led to limited accuracy of their imputation.

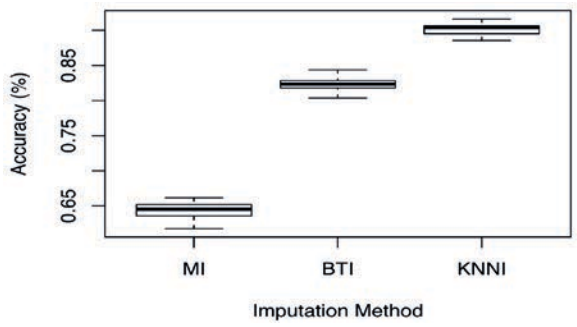


**Figure 1**. Label imputation for MI, BTI and KNNI over 30 runs for 2 classes. Each boxplot displays the minimum and maximum values (whiskers), the first and the third quartile (boundaries of the box represent 50% of the data), and the median (the thick line)

In the 11-class case (Figure 2), the effect is even more evident, as the samples are not equally distributed among the classes, leading to a low accuracy for MI and BTI and a high variance for KNNI due to the randomness of the labels removed in each of the 30 runs.

### 6.2 Continuous Feature Imputation

The second experiment aimed to validate the algorithms' accuracy in presence of continuous features. Eight features from the dataset without missing data (RFmi, RFma, PRImi, PRIma, PDmi, PDma, SPmi, SPma) were considered in this imputation. As in the previous experiment, 25% of

**Table 4**. Sample subset with imputed radar data and natural number coding of 'RFC', 'PRC', 'PDC', and 'ST'

| ID | RFC | RFmi | RFma | PRC | PRImi | PRIma | PDC | PDmi | PDma | ST | SPmi | SPma |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 84 | 2 | 5300 | 5800 | 7 | 963.2 | 5625 | 1 | 5.8 | 17 | 1 | 5.9 | 6.1 |
| 4354 | 4 | 2700 | 2900 | 4 | 1351 | 1428 | 1 | 4 | 6.3 | 1 | 9.5 | 10.5 |
| 7488 | 2 | 8800 | 9300 | 7 | 100 | 125 | 1 | 13 | 21 | 2 | 1.4 | 1.6 |
| 9632 | 4 | 137 | 139 | 11 | 622.6 | 31312 | 2 | 61.1 | 93.1 | 4 | 12 | 47.8 |
| 9839 | 9 | 2900 | 3100 | 6 | 2058 | 48128 | 2 | 99 | 101 | 1 | 9.5 | 10.5 |

**Table 5**. Example of binary coding for 32-level categorical variable

| Original Category | | Encoded Variables | | | | |
|---|---|---|---|---|---|---|
| **Index** | **Label** | **B1** | **B2** | **B3** | **B4** | **B5** |
| 1 | '2D' | 0 | 0 | 0 | 0 | 0 |
| 2 | '3D' | 0 | 0 | 0 | 0 | 1 |
| 3 | 'AA' | 0 | 0 | 0 | 1 | 0 |
| ... | | | | | | |
| 16 | 'CS' | 0 | 1 | 1 | 1 | 1 |
| ... | | | | | | |
| 32 | 'ME' | 1 | 1 | 1 | 1 | 1 |

**Table 6**. Example of dummy coding for 32-level categorical variable

| Original Category | | Encoded Variables | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Index** | **Label** | **D1** | **D2** | **D3** | **D4** | **D5** | ··· | **D16** | ··· | **D32** |
| 1 | '2D' | 1 | 0 | 0 | 0 | 0 | ··· | 0 | ··· | 0 |
| 2 | '3D' | 0 | 1 | 0 | 0 | 0 | ··· | 0 | ··· | 0 |
| 3 | 'AA' | 0 | 0 | 1 | 0 | 0 | ··· | 0 | ··· | 0 |
| ... | | | | | | | | | | |
| 16 | 'CS' | 0 | 0 | 0 | 0 | 0 | ··· | 1 | ··· | 0 |
| ... | | | | | | | | | | |
| 32 | 'ME' | 0 | 0 | 0 | 0 | 0 | ··· | 0 | ··· | 1 |

the data was removed for each feature and subsequently imputed. The Root Mean Square Error (RMSE) between the imputed values and the real ones was selected to measure the methods' performance. The experiment was iterated 30 times and the non-parametric Wilcoxon test for statistical significance [30] along with the relative effect size (Cohen's $d$) [31], were calculated.

The Wilcoxon test uses the rank of the data to determine if there is any difference between two samples, without making any assumption about the data distributions' nature. If the *p-value* is greater than the significance threshold $\alpha$ ($\alpha$ = 0.05 in our case), then there is no significant difference between the two samples.

The Cohen's $d$ effect size metric shows how much, on average, one technique outperforms another. The measure applied to the two populations ($1^{st}$ technique vs $2^{nd}$ technique) gives a response between 0 and 1 and is calculated with (1)

$$d = \frac{\left|\mu^2_{group1} - \mu^2_{group2}\right|}{|\sigma_{both}|}, \qquad \sigma_{both} = \sqrt{\frac{\sigma^2_{group1} + \sigma^2_{group2}}{2}}, \tag{1}$$

where $\mu_{group1}$ and $\mu_{group2}$ are the average of the two techniques respectively, and $\sigma_{both}$ is the average of their standard deviations.

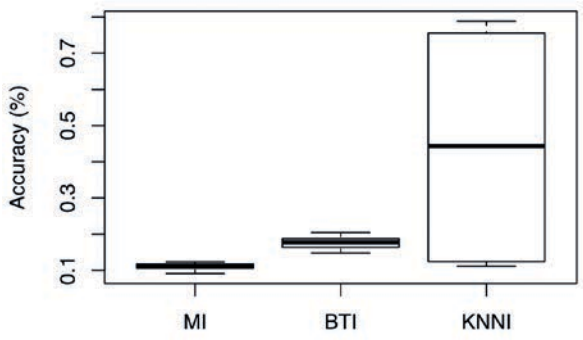


**Figure 2**. Label imputation with MI, BTI and KNNI over 30 runs for the 11 classes. Each boxplot displays the minimum and maximum values (whiskers), the first and the third quartile (boundaries of the box represent 50% of the data), and the median (the thick line)

In [31], the results are grouped in 3 categories: small with $d \in [0.2, 0.5)$; medium $d \in [0.5, 0.8)$; and large $d \in [0.8, 1.0]$. If d<0.2 the difference of the two groups is insignificant, even if the *p-value* shows statistical significance.

The considered imputation algorithms were also compared with the median imputation (an algorithm from the "single imputation" family, largely rejected by the research community) and for this reason, used just as a comparison baseline. As expected (Figure 3), all other algorithms produced smaller errors, outperforming the median imputation. In 23 out of 25 cases BTI had better statistical significance and median (*p-value* $< 0.025$) than MI and KNNI and with effect size $d$=1. The comparison between BTI and KNNI on PRImi and PRIma, despite a *p-value* $< 0.025$, showed slightly lower effect size, $d$=0.83 and $d$=0.78 respectively. Analysing the results for the two features led to the conclusion that KNNI is again strongly influenced by the data distribution, which is highly positively skewed for PRImi and PRIma (Figure 4).

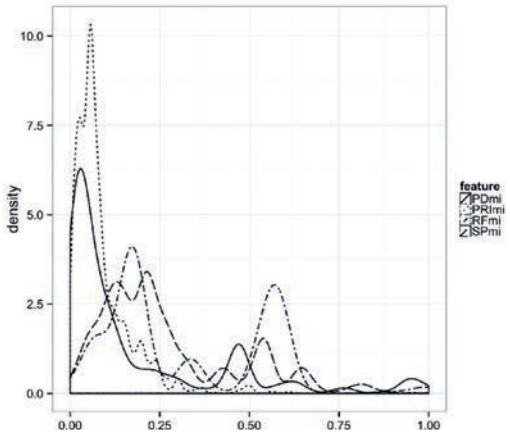


**Figure 4**. Density function for the features: RFmi, PRImi, PDmi, SPmi on the dataset without missing value. Each feature is normalized within (0, 1) interval. The distribution of PRImi and PDmi sows highly positive skewedness. For a better visualisation, the features RFma, PRIma, PDma and SPma are omitted due to the similar distribution and high correlation with the respective minima

Figure 4 shows the normalized density function for four features, from which PRImi and PDmi have high positive skewness, concentrated in the first quartile. It seems, as for the label imputation,

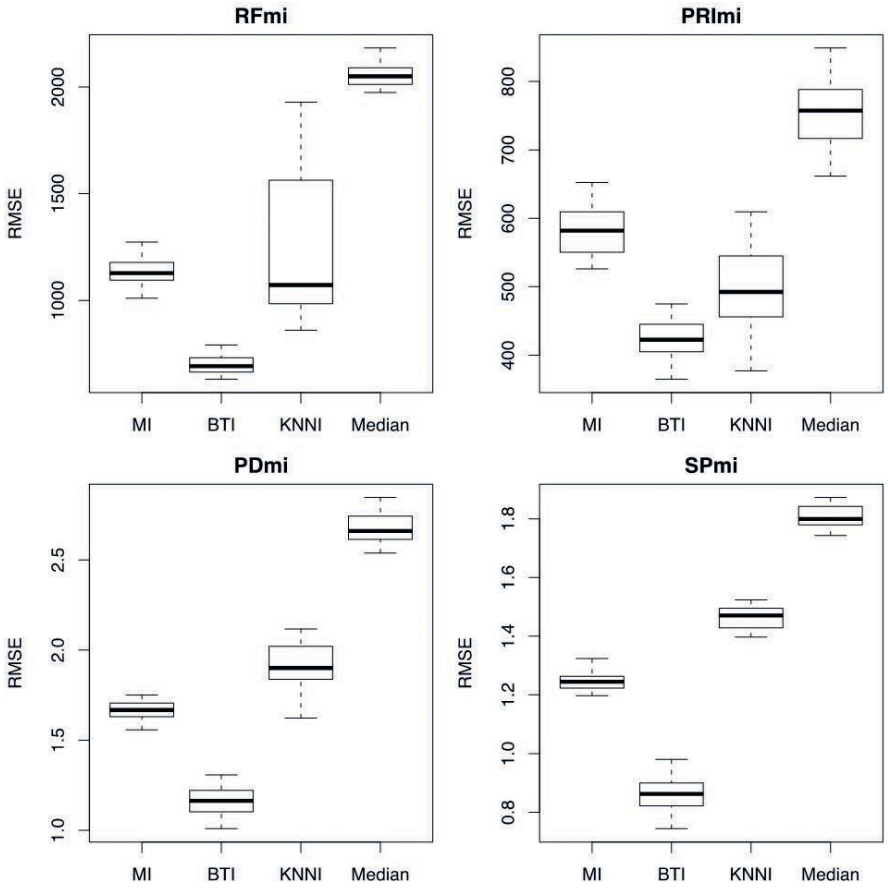


**Figure 3**. Root mean square error (RMSE) for the imputation of the continuous values (30 runs). Low median values represent preferable imputation methods. Non-overlapped boxplots indicate statistical difference between the algorithms. For a better visualisation, the features RFma, PRIma, PDma and SPma have been omitted due to the similar distribution and high correlation with the respective minimum

that a highly imbalanced distribution can affect the imputation process, explaining why each imputation method produces different accuracy for the features given in Figure 3 (although, BTI is the overall winner).

The results for the Wilcoxon test subjected to the Benjamini-Hochberg and Bonferroni corrections [32] for multiple statistical test are still significant ($\alpha = 0.05$).

## 7 Classifiers Training Results

The efficiency of each classifier when solving the radar emitter recognition problem is tested with two main experiments. The dataset is split into two subsets before the imputation: training one (75% of the whole data) and testing one (25% of the whole data).

Batch-mode training is adopted and the investigated neural network topologies include one hidden layer with fully connected neurons between the adjacent layers. For a given experiment with $P$ learning samples, the error function is given with (2)

$$E_p = \frac{1}{2}\sum_{p=1}^{P}\sum_{i=1}^{L}(x_i^p - t_i^p)^2 \qquad (2)$$

where for each sample $p=1,\ldots, P$ and each neuron of the output layer $i=1,\ldots, L$; a pair $(x_i, t_i)$ of NN output and the target values is defined respectively. NN learning with *Levenberg-Marquardt* algorithm is then carried out and the training set is further divided into 80% for the training and 20% for the validation. Mean Squared Error (MSE) is used for evaluating the NN learning performance. The stopping criterion is set to 500 training epochs, or gradient reaching value less than 1.0e-06, or 6 consequent failed validation checks, whichever occurs first.

The basic split of the data (75% training, 25% testing) is also used in the RF classification case. The limit for the forests is set to 500 trees and the output class is decided by a vote among them.

The SVM are provided with a radial basis kernel (other kernels were tried as well but led to worse results) and the input parameters are generated randomly by the algorithm. Moreover, we also used parameters optimization function ("tune.svm") from the package, which increased the classifier accuracy with up to 4%.

For the first experiment, the NN topology is N-N-2, where N is the number of inputs and the output contains 2 binary neurons codded as: 10 for class *"Civil"*; and 01 for class *"Military"*. For the second experiment, the same topology is used with 11 output neurons (representing 4 *civil* and 7 *military* classes).

After coding the categorical variables with the three methods described above, the accuracy of each classifier is investigated, evaluated and compared before and after the data normalisation and standardisation.

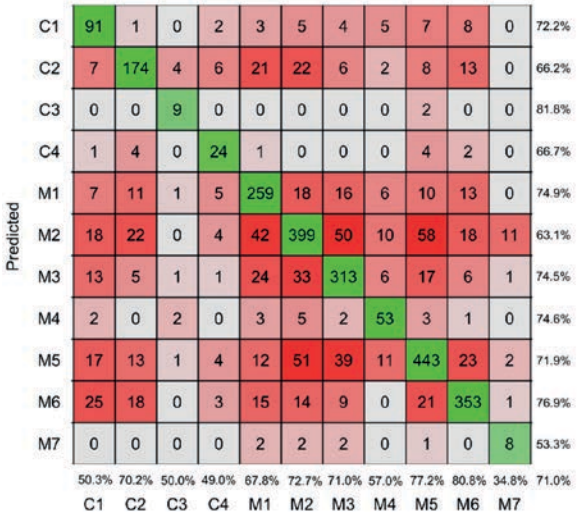


**Figure 6**. Confusion matrix illustrating the RF classification results for the 11 classes, after employing BTI with continuous value coding. The batch includes 7 military ('M1' – Multi-function, 'M2' – Battlefield, 'M3' – Aircraft, 'M4' – Search, 'M5' – Air Defence, 'M6' – Weapon and 'M7' – Info) and 4 civil classes ('C1' – Maritime, 'C2' – Airborne Navigation, 'C3' – Meteorological, 'C4' – Air Traffic Control)

Sample confusion matrices are shown in Figure 5 and Figure 6 for the best accuracy achieved in the experiments, after training with continuous input data. The RF demonstrates high accuracy in both of them (the number of correct responses given in the green squares and the number of incorrect ones in the red squares). The bottom right percentage illustrates the overall classifier accuracy (OCA). Furthermore, it can be observed from Figure 6 that the number of hits, as well as the accu-

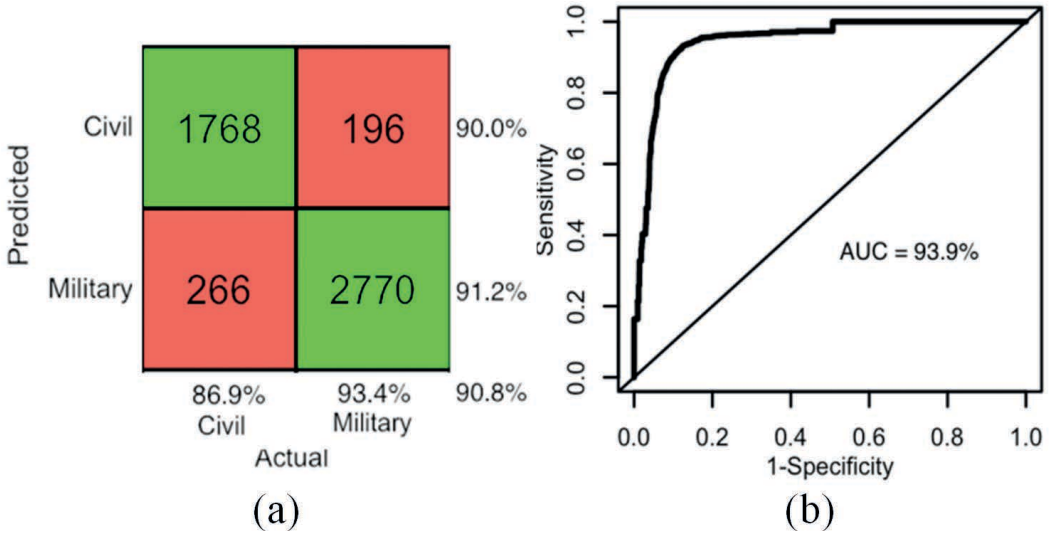


**Figure 5**. **(a)** Confusion matrix illustrating the RF classification results for the two classes, after employing BTI with continuous value coding (*Military* and *Civil*). **(b)** ROC Curve for the same simulation

racy of the RF classifier, compared to the previous work [24], is increased. Another important achievement is also illustrated in these figures: the class accuracy variance is now within the 34.8% to 90.8% interval; while in [24] it was between 22.6% and 87.4%. This may be attributed to the higher number of available training and testing samples as a result of the BTI imputation, which increased the used dataset statistical power and improved the classification performance of the RF.

The results shown in Figure 7 and Figure 8, illustrate moderate impact of the categorical coding on the classification, while the continuous coding appears to be more efficient (1% better for the 2-class case, and 5% for the 11 classes). For the two classes (Figure 7), the best result (90.80%) is obtained when combining the RF classifier with the BTI and continuous values; and for the eleven classes, the same combination achieved again the best accuracy of 71.0% (Figure 8).

In Figure 7 and Figure 8, the SVM columns (the last three columns) have the same results for the scaled and standardized data, since the algorithm performs internally the two operations before the classification.

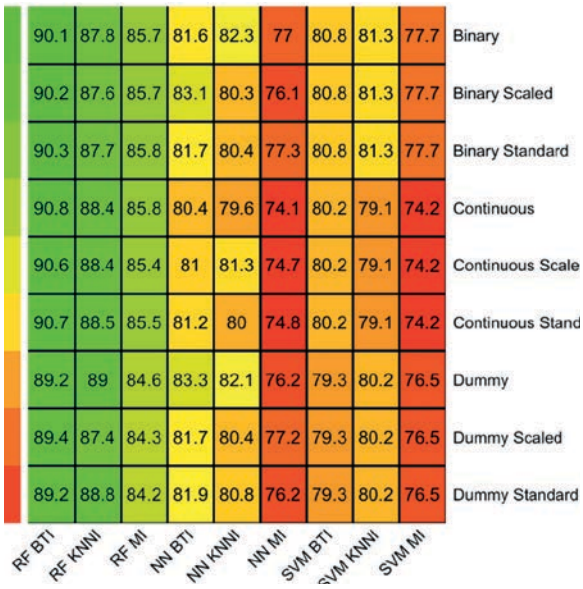


**Figure 7**. Classification performance results of the classifiers (RF, NN and SVM) in the case of 2 classes after employing three different imputation techniques (BTI, KNNI and MI), for the three different groups of coding (binary, continuous and dummy). The colour scale to the left shows the achieved accuracy percentile

In 98 out of 108 comparisons, BTI has demonstrated the best accuracy for all classifiers, while the KNNI achieved best results in 8 cases and MI in 2 comparisons only.

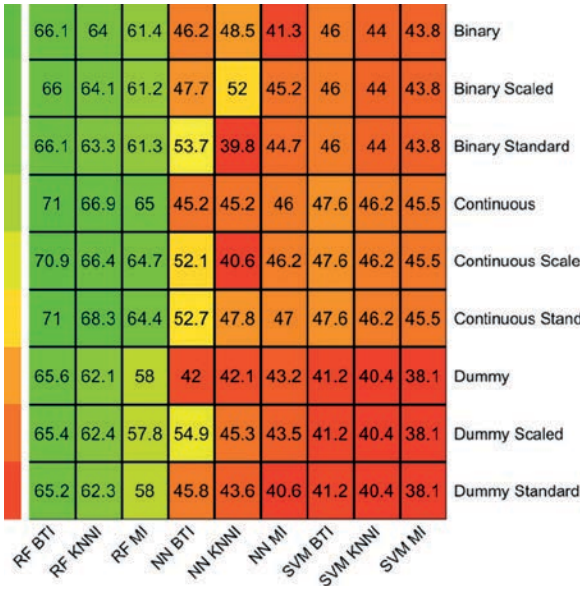


**Figure 8**. Classification performance results of the classifiers (RF, NN and SVM) in the case of 11 classes after employing three different imputation techniques (BTI, KNNI and MI), for the three different groups of coding (binary, continuous and dummy). The colour scale to the left shows the achieved accuracy percentile

Taking closer look at the confusion matrix given in Figure 6, it can be seen that the number of misclassified samples within the classes of the same family (e.g., 'civil') is higher than the number of misclassified samples as belonging to the other family classes (e.g., 'military'). For example, if we consider the M1 class ($5^{th}$ row in Figure 6), 24 samples of M1 class are wrongly classified as belonging to civil classes (C1 to C4), while 63 samples are mislabelled as of the other military classes (M2 to M7). It is even more evident from the last row – M7 class, for which all misclassified samples (seven) belong to the military family.

Since it is not possible to give a cost for the misclassification of each class, a ROC curve analysis for multiclass problems is used to assess the classifiers accuracy under different conditions [33], [34].

Let's call a superclass the union of all classes belonging to the same general type or family ('civil' or 'military') for the 11-class problem: C = U($C_1$,..., $C_4$); and M = U($M_1$,..., $M_7$). We also define two types of misclassification errors: outer error (OE) and inner error (IE).

In the 11-class case, the OE occurs when: one of the civil samples belonging to a $C_i$, i =1,.., 4, class is misclassified as belonging to a military one $M_j$, j= 1,..,7; or when a military sample is mislabelled as a civil one. An IE occurs when a civil sample is misclassified as belonging to another civil class or when a military pattern is mislabelled with a different military label.

Let us now denote the Inner Accuracy (IA) as the accuracy obtained calculating the ROC curve for a multi-class problem, as proposed in [34], when applied only on classes belonging to the same superclass. The area under the curve (AUC) is obtained by averaging the AUC of all considered pairwise classes. This approach measures how well each class is separated from the others, emphasising that certain pairs of classes can be well separated, even when the superclasses cannot be well separated.

Labelling the civil classes $C_1$, ..., $C_k$, (k = 4), we estimate the probability of each test sample x belonging to any class $C_i$ as: p($C_i$ |x), for i = 1, ..., k. For any pair of classes ($C_i$, $C_j$), it is possible to compute measure 'A' using p($C_i$|x) or p($C_j$ |x), hence, A($C_i$ |$C_j$) is the probability of a randomly selected member of class $C_i$ to have lower estimated probability of belonging to class $C_j$, than a randomly selected member of class $C_j$.. Because for a 2 class problem (class 0 and class 1): A(0 |1) = A(1 |0); and for a multi-class problem A($C_i$|$C_j$) ≠ A($C_j$|$C_i$), we use the average A($C_i$, $C_j$) = (A($C_i$ |$C_j$) + A($C_j$ |$C_i$) )/2, adopted as measure of separability between $C_i$ and $C_j$, the overall performance of the classification in separating $k$ classes is then the average of this measure over all class pairs

$$IA = \frac{2}{k(k-1)} \sum_{i<j} A(C_i, C_j). \qquad (3)$$

The difference 1-IA represents on average, the percentage of wrongly labelled patterns in the same superclass.

On the other hand, the Outer Accuracy (OA) is calculated applying the average AUC to all pairs ($C_i$, $M_j$)

$$OA = \frac{\sum A(C_i, M_j)}{kn}, \qquad (4)$$

where $i=1,..., k$, $j=1,..., n$, and $k$ and $n$ are the number of civil and military classes respectively. Again, the difference 1-OA represents the percentage of patterns, misclassified as belonging to the other superclass.

**Table 7**. Classifiers inner accuracy (IA) and outer accuracy (OA) compared to the best overall accuracy (OCA) in the 11 class classification (given in figure 8) for RF, NN and SVM; best OA for RF, NN, SVM; and results for two classifiers with same OCA but different OA

| CLASSIFIER | IA | OA | OCA |
|---|---|---|---|
| RF BTI continuous (RFbest OCA) | 87.1 | 88.7 | 71.0 |
| RF KNNI continuous standardised (RFbest OA) | 89.1 | 90.6 | 68.3 |
| NN BTI dummy scaled (NNbest OCA) | 74.1 | 74.2 | 54.9 |
| NN BTI continuous scaled (NNbest OA) | 81.0 | 84.3 | 52.7 |
| SVM BTI continuous (SVMbest OCA) | 64.8 | 67.7 | 47.6 |
| SVM KNNI continuous scaled (SVMbest OA) | 62.9 | 69.0 | 46.2 |
| NN MI continuous | 75.6 | 78.8 | 46.0 |
| SVM BTI binary standardised | 64.9 | 67.7 | 46.0 |

Results for IA, OA and OCA are given in Table 7, where in the first six rows the best OA and OCA accuracies for the RF, NN and SVM classifiers are shown (which performance is illustrated with Figure 8). The last two rows display the results for NN (with MI and continuous coding) and SVM (with BTI, binary coding and standardisation).

The IA and OA can help to better understand the underlying distribution of the misclassified samples among the classes. Looking at the results in Figure 8, it is evident that for some of them, it is difficult to choose the best classifier due to the similar OCA values. In such cases the OA can provide insightful information and help identify the best one. For example, the OCA is identical (46%) for the NN (with MI and continuous coding) and SVM (with BTI, binary coding and standardization) classifiers. Nevertheless, looking at the OA column in table 7 (last 2 cells), one can see that the NN classifier is with 11.1% better accuracy than the SVM one.

In theory, it may be speculated that the probability of having a misclassified sample within the same superclass is higher (since they are coming from the same family of radar signals), than the probability of a class being misclassified as belonging to the other superclass. Considering the values given in Table 7, it can be observed that the IA is generally lower than the OA, which is in agreement with the above presumption. Moreover, the RF, NN and SVM classifier configurations producing the best OCA (71%, 54.9%, and 47.6%: rows 2, 4, and 6, respectively) are not the best ones when OA is used as a metric (then their OCA are 68.3%, 52.7%, and 46.2%: rows 3, 5, and 7, respectively). This enforces our point that a second metric should be used when assessing classifiers performance. For example, the RF classifiers with the highest OCA and OA are shown in the first 2 rows of Table 7. In the first case ($1^{st}$ row – (RFbest OCA)), 29% of the patterns are misclassified (1450 of 5000) and 11.3% of them are assigned to the wrong superclass (164 patterns). In the second case ($2^{nd}$ row - (RFbest OA)), 31.7% of the samples (1680 of 5000) are wrongly labelled and 9.4% of them (158) are assigned to the wrong superclass.

As for the NN classifiers ($3^{rd}$ and $4^{th}$ rows), the difference in the OCA is very small (2.2%), with 2255 and 2365 misclassified samples respectively, while the relative OA difference is substantial - 10.1%, with 582 and 371 misclassified patterns respectively. Similar results can be observed for the SVM classifier as well ($6^{th}$ and $7^{th}$ rows). All these results lead to the conclusion that the use of OA can provide extra insight about the classifiers' performance, giving additional metric for choosing the best classification model configuration (considering the trade-off between OCA and OA).

# 8 Conclusion

This work shows that the performance of the employed classifiers for recognition and identification of radar sources, based on datasets of radar signal characteristics depends a lot on the quality of the available data and dealing with the missingness in the datasets can improve the classifier overall accuracy. For this purpose, three different model based approaches for imputation are investigated here for substituting large number of missing data (up to 60% missingness). The implementation of

MI, BTI, and KNNI models tripled the number of samples with completed values, which in turn led to improved results for the classifiers. The performance of the imputation methods is assessed with Wilcoxon's test for statistical significance (showing no significant difference between the substituted samples) and Cohen's effect size metric to assess how much, on average, one technique outperforms another.

The use of Bagged Tree imputation method triggered best accuracy for all supervised classifiers (especially in the 11-class case). Once the missingness in the datasets is dealt with, two case studies are investigated: a 2-class and 11-class classification and neural networks, support vector machines and random forests are implemented for solving these two tasks. In the 2-class case, when two superclasses (military and civil) are investigated, the best performance was achieved by the RF after BTI imputation and continuous coding (90.8%), followed by the NN (83.3%) and SVM (81.3%). These results are in agreement with Weinberg et al. work [23], who argued that random forests do not have significantly higher percent accuracy than support vector machines and neural networks, contradicting the Fernandez-Delgado et al. paper [22], where they claim that 'RF is the best family classifier'. In our view, what can be claimed for sure is that each classifier accuracy depends considerably on the pre-processing phase (the difference best–worst performance for each classifier investigated in this work was between 6%-9% in the two-class case, and between 10%-14% in the 11-class case), and obviously the assessment of their performances depends on the metric used for the comparison. In the second case study, the two superclasses are further specialised into four civil and seven military classes – defining an 11-class problem. We use the ROC curve analysis for a multi-class problem, and obtain the area under the curve (AUC) by averaging the AUC of all considered pairwise classes. This approach shows that some pairs of classes can be well separated, even when the superclasses are not well distinguished.

We also show that the introduced two new metrics: inner (super) class accuracy IA (eq. 3); and the outer class accuracy OA (eq. 4), can be used to complement the overall classifier accuracy (OCA) metric when choosing the best classifier configuration for the problem at hand. Especially in cases when the OCA has similar values for the classifiers, the OA should be used as an additional criterion for the choice of the most efficient and accurate classification model. This investigation found that the IA is usually smaller than the OA, which is in agreement with the presumption that it is more likely a class to be misclassified as one from the same superclass (same family), than to be misclassified as belonging to the other superclass. Future work may include identification of more than two superclasses by implementing unsupervised learning and then employing supervised one for assessing IA and OA metrics.

## References

[1] C. Enders, Applied missing data analysis. Guilford Press, New York, 2010.

[2] J. Osborne, Best Practices in Data Cleaning. SAGE, 2013.

[3] P. Schmitt, J. Mandel, M. Guedj, A Comparison of Six Methods for Missing Data Imputation. Journal of Biometrics & Biostatistics, 6(1), 2015, 1-6.

[4] G. Ridgeway, Generalized Boosted Models: A guide to the gbm package. Update 1.1, 2007. www.saedsayad.com/docs/gbm2.pdf. Accessed 20 October 2016.

[5] M. Richards, Fundamentals of radar signal processing. Tata McGraw-Hill Education, 2005.

[6] I. Jordanov, N. Petrov, Intelligent Radar Signal Recognition and Classification. In Abielmona, R., Falcon, R., Zincir-Heywood, N., Abbass, H. (eds.) Recent Advances in Computational Intelligence in Defense and Security, 2016, 101-135.

[7] I. Jordanov, N. Petrov, A. Petrozziello, Supervised radar signal classification. Neural Networks (IJCNN), 2016 International Joint Conference on. IEEE., 2016, 1464-1471.

[8] L. Carro-Calvo, et al., An evolutionary multiclass algorithm for automatic classification of high range resolution radar targets. Integrated Computer-Aided Engineering, 16(1), 2009, 51-60.

[9] E. Granger, M. Rubin, S. Grossberg, P. Lavoie, A What-and-Where fusion neural network for recognition and tracking of multiple radar emitters. Neural Networks, 14 (3), 2001, 325-344.

[10] S. Maytal, F. Provost, Handling missing values when applying classification models. Journal of Machine Learning Research, 8, 2007, 1625-1657.

[11] N. Ibrahim, R. Abdullah, M. Saripan, Artificial neural network approach in radar target classification. Journal of Computer Science, 5(1), 2009, 23.

[12] M. Ahmadlou, H. Adeli, Enhanced probabilistic neural network with local decision circles: A robust classifier. Integrated Computer-Aided Engineering, 17(3), 2010, 197-210.

[13] Z. Yin, W. Yang, Z. Yang, L. Zuo, H. Gao, A study on radar emitter recognition based on SPDS neural network. Information Technology Journal, 10(4), 2011, 883-888.

[14] M. Gong, J. Zhao, J. Liu, Q. Miao, L. Jiao, Change Detection in Synthetic Aperture Radar Images Based on Deep Neural Networks, IEEE Trans. on Neural Networks and Learning Systems, 27(1), 2016, 125-138.

[15] C. Shieh, C. Lin, A vector neural network for emitter identification. IEEE Trans. on Antennas and Propagation, 50(8), 2002, 1120-1127.

[16] S. Zhai, T. Jiang, A new sense-through-foliage target recognition method based on hybrid differential evolution and self-adaptive particle swarm optimization-based support vector machine, Neurocomputing, 149(1), 2015, 573-584.

[17] Z. Xin, W. Ying, Y. Bin, Signal classification method based on support vector machine and high-order cumulants. Wireless Sensor Network, 2(1), 2010, 48-52.

[18] E. Abdulkadir, I. Onaran, Pulse Doppler radar target recognition using a two-stage SVM procedure. Aerospace and Electronic Systems, 47(2), 2011, 1450-1457.

[19] A. Karatzoglou, M. David, H. Kurt, Support vector machines in R, Department of Statistics and Mathematics, WU Vienna University of Economics and Business, 2005.

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

[21] A. Yali, D. Geman, Shape quantization and recognition with randomized trees. Neural computation, 9(7), 1997, 1545-1588.

[22] M. Fernandez-Delgado, E. Cernadas, S. Barro, D. Amorim, Do we need hundreds of classifiers to solve real world classification problems? Journal of Machine Learning Research, 15(1), 2014, 3133-3181.

[23] M. Wainberg, B. Alipanahi, B. Frey, Are Random Forests Truly the Best Classifiers? Journal of Machine Learning Research 17, 2016, 1-5.

[24] I. Jordanov, N. Petrov, Sets with Incomplete and Missing Data – NN Radar Signal Classification. IEEE WCCI'14 World Congress on Computational Intelligence, Beijing, China, 2014, 218-225.

[25] R. Geaur, Z. Islam, A decision tree-based missing value imputation technique for data pre-processing. Proceedings of the Ninth Australasian Data Mining Conference, 121, 2011, 41-50.

[26] A. Feelders, Handling missing data in trees surrogate splits or statistical imputation? Principles of Data Mining and Knowledge Discovery. Springer Berlin Heidelberg, 2009, 329-334.

[27] A. Petrozziello, I. Jordanov, Data Analytics for Online Travelling Recommendation System: A Case Study. Proceedings of the IASTED International Conference Modelling, Identification and Control (MIC 2017), Innsbruck, Austria, 2017, 106-112.

[28] M. Templ, A. Kowarik, P. Filzmoser, Iterative stepwise regression imputation using standard and robust methods. Journal of Computational Statistics and Data Analysis, 55, 2011, 2793-2806.

[29] S. Verboven, K. Branden, P. Goos, Sequential imputation for missing values. Computational Biology and Chemistry, 31(5), 2007, 320-327.

[30] F. Sarro, A. Petrozziello, M. Harman, Multi-objective software effort estimation. Proceedings of the 38th International Conference on Software Engineering, ACM, 2016, 619-630).

[31] J. Cohen, Statistical power analysis for the behavioural sciences. Routledge, New York, 2013.

[32] P. Dalgaard, Introductory Statistics with R. Springer, New York, 2008.

[33] J. Huang, C. Ling, Using AUC and accuracy in evaluating learning algorithms. IEEE Transactions on Knowledge and Data Engineering, 17(3), 2005, 299-310.

[34] D. Hand, R. Till, A simple generalisation of the area under the ROC curve for multiple class classification problems. Machine learning, 45(2), 2001, 171-186.

**Ivan Jordanov** received his PhD degree in computer aided optimization of dynamic systems and MSc in applied mathematics and informatics from the Technical University of Sofia (Bulgaria). He is currently a Reader at the School of Computing, University of Portsmouth, UK. His research interests include computational intelligence (neural networks, heuristic global optimization, and data analytics) for solving pattern recognition and classification problems. He is an associate editor of two international journals, co-author of five textbooks, editor of two Springer-Verlag books, and his publication list includes more than 80 papers in peer-reviewed journals and conference proceedings.

**Nedyalko Petrov** received his PhD in intelligent methods for pattern recognition and optimization from the University of Portsmouth, UK. Dr Petrov has vast experience in conducting research and applying computational intelligence, statistical and data analysis techniques for addressing real-world problems, both in academia and industry (including short KTPs with BP and Airbus). His main research interests include machine learning, mathematical modelling and optimization, pattern recognition and classification. Currently, Nedyalko Petrov is a Senior Bioinformatician at the Comprehensive Biomedical Research Centre (BRC) at GSTT / King's College London, where he conducts research and analysis of complex medical data sets (including identification of new biological populations, discovering differences between patients' samples and drug responses).

**Alessio Petrozziello** is presently a PhD candidate at the University of Portsmouth (School of Computing) with subject "Machine Learning Methods for Big Data." In 2013 he was granted a full scholarship issued by the Italian Ministry of Education, University and Research (MIUR) which allowed him to spend a semester at the Penn State University, doing research on machine learning techniques for the identification of natural hazards. He also spent the Summer 2016 as a Data Scientist Intern with Expedia travel company, working on recommender systems for big data. He holds an BSc in Computer Science from the University of Salerno, Italy. His key qualications and areas of interest include data mining, deep learning, big data analytics and recommender systems.