Machine Learning: Science and Technology



# Feature space reduction method for ultrahigh-dimensional, multiclass data: random forest-based multiround screening (RFMS)

IOP Publishing Mach. Learn.: Sci. Technol. 4 (2023) 045012 https://doi.org/10.1088/2632-2153/ad020e

MACHINE LEARNING
Science and Technology



OPEN ACCESS

RECEIVED
26 June 2023

REVISED
20 September 2023

ACCEPTED FOR PUBLICATION
10 October 2023

PUBLISHED
19 October 2023



PAPER

# Feature space reduction method for ultrahigh-dimensional, multiclass data: random forest-based multiround screening (RFMS)

**Gergely Hanczár[1], Marcell Stippinger[2], Dávid Hanák[1], Marcell T Kurbucz[2,3,*], Olivér M Törteli[1], Ágnes Chripkó[1] and Zoltán Somogyvári[2]**

[1] Cursor Insight Ltd, 20-22 Wenlock Road, N17GU London, United Kingdom
[2] Department of Computational Sciences, Wigner Research Centre for Physics, 29-33 Konkoly Thege Miklós Street, H-1121 Budapest, Hungary
[3] Institute of Data Analytics and Information Systems, Corvinus University of Budapest, 8 Fővám Square, H-1093 Budapest, Hungary
* Author to whom any correspondence should be addressed.

**E-mail:** kurbucz.marcell@wigner.hun-ren.hu

**Keywords:** feature screening, ultrahigh dimensionality, multiclass classification, random forest, biometrics

## Abstract

In recent years, several screening methods have been published for ultrahigh-dimensional data that contain hundreds of thousands of features, many of which are irrelevant or redundant. However, most of these methods cannot handle data with thousands of classes. Prediction models built to authenticate users based on multichannel biometric data result in this type of problem. In this study, we present a novel method known as *random forest-based multiround screening (RFMS)* that can be effectively applied under such circumstances. The proposed algorithm divides the feature space into small subsets and executes a series of partial model builds. These partial models are used to implement tournament-based sorting and the selection of features based on their importance. This algorithm successfully filters irrelevant features and also discovers binary and higher-order feature interactions. To benchmark RFMS, a synthetic biometric feature space generator known as *BiometricBlender* is employed. Based on the results, the RFMS is on par with industry-standard feature screening methods, while simultaneously possessing many advantages over them.

## 1. Introduction

The understanding of human motor coordination and the building of prediction models to meet various business needs have become widely studied topics in fields such as neurology and cybersecurity. With the help of adequate sensors, gestures, walking, handwriting, eye movement, or any other human motor activity can be transformed into a multidimensional time series. However, in general, any fixed set of features is either not representative of these time series or too large for resource-efficient classification. Thus, instead of computing an *a priori* defined, conveniently small set of features, a promising alternative strategy is to create an ultrahigh-dimensional dataset that consists of hundreds of thousands of features and search for the most informative minimal subset [1]. In this process, as well as in many other machine learning (ML) applications, the evaluation of feature importance and the elimination of irrelevant or redundant predictors have become crucial elements in improving the performance of algorithms [2]. This elimination can increase the accuracy of the learning process and reduce the resource needs of model building. The statistical challenges of high dimensionality have been thoroughly reviewed in [3].

Traditional variable selection methods do not usually work well in ultrahigh-dimensional data analysis because they aim to specifically select the optimal set of active predictors [4]. It has also been reported that traditional dimensionality reduction methods, such as principal component analysis (PCA), do not yield satisfactory results for high dimensional data (for example, see [5, 6]). In contrast to these methods, feature

screening uses rough but fast techniques to select a larger set that contains most or all of the active predictors [7]. Although several screening methods have been published for ultrahigh-dimensional data in recent years, only a few of them can be used in cases when the response variable contains numerous classes. In particular, in the domains of neuroscience and biometric authentication, datasets with these properties are often encountered.

To reduce various ultrahigh-dimensional feature spaces in binary classification problems, Fan and Lv [8] proposed a sure independence screening method in the context of linear regression models with thousands of features and only hundreds of samples to generalize from. This paper introduced the concept of *sure screening,* which means that all the important variables survive after variable screening with probability tending to one. According to Fan and Fan [9], all features that effectively characterize both classes can be extracted by using two-sample $t$-test statistics, resulting in features annealed independence rules, however, the $t$-tests evaluate the features one-by-one without considering feature interactions. Mai and Zou [10] used a Kolmogorov filter (KF) method, which is also applied for the ultrahigh-dimensional binary classification problem with a dependent variable in Lai *et al* [11]. Roy *et al* [12] proposed a model-free feature screening method based on energy distances (see [13, 14]). Note that all the papers cited in this paragraph deal with binary classification only.

Mai and Zou [15] extended the KF method to handle multiclass response, however, the simulated datasets used in the paper had only about 5000 features and a couple hundred samples. Ni *et al* [16] proposed adjusted Pearson chi-square feature screening based on weighting for multiclass classification, an approach that estimates the information value of the features one-by-one, ignoring potential interactions. Ni and Fang [17] applied information entropy theory to model-free feature screening for ultrahigh-dimensional, multiclass classification. Their method introduces categorical covariates, but also ignores interactions. Soft computing techniques, in particular meta-heuristic approaches such as genetic algorithms (GAs) and particle swarm optimization (PSO) have also found application in this domain. Hybrid techniques combining GAs with an elastic net embedded method [18], binary GAs with feature granulation [19], and competitive swarm optimization [20] have shown promise in addressing the complexity of feature screening. While these approaches do account for feature interactions, the evaluated datasets include up to 5000 features only. Harnessing fuzzy cost-based feature selection through interval multi-objective PSO [21] is directed toward addressing high-dimensional data challenges, however, it focuses on minimizing the *cost* of retrieving the necessary features, rather than finding the few relevant covariates. Evolutionary feature subset selection grounded in interaction information [22] and the application of self-adaptive PSO [23] are also notable results in this area, but the former method was evaluated on datasets with only two classes in the cases when the number of features was high enough, and the latter was only shown to work on UCI ML repository [24] databases with up to no more than 6400 features and 26 classes. Moreover, Saadatmand and Akbarzadeh [25] introduced the set-based integer-coded fuzzy granular evolutionary (SIFE) algorithm, applicable to ultrahigh-dimensional, multiclass feature spaces, but it was only evaluated on datasets with ten classes at most.

While most existing feature screening approaches are unsuitable for examining higher-order interactive structures and nonlinear structures, random forest (RF) [26] can overcome such difficulties [27]. To provide a robust screening solution for ultrahigh-dimensional, multiclass data, we propose the *RF-based multiround screening (RFMS)* method. The Julia package that implements RFMS is publicly available on GitHub [28]. The RFMS improves the accuracy and scalability of both traditional selection methods and existing RF-based screening by organizing the screening process into rounds. As an advantage, the input is processed in larger chunks, and we can iteratively distill a well-predicting subset of features.

The main contributions of the paper are outlined as follows:

- Introduction of the RFMS method, a novel approach for feature screening in ultrahigh-dimensional, multiclass datasets. RFMS efficiently selects informative features and feature combinations (accounting for feature interactions), even when the features are largely irrelevant, and the information content in any relevant feature is relatively small. This makes RFMS particularly suitable for domains with large and complex datasets, such as biometric authentication.
- Comprehensive benchmarking analysis of RFMS against established feature screening methods such as PCA, factor analysis (FA), and $k$-best screening. The benchmarking employs three basic classifiers on synthetic data emulating real signature datasets.
- Availability of the RFMS implementation through a Julia package on GitHub, fostering reproducibility and further advancement of the proposed method.

The paper is organized as follows. Section 2 introduces the dataset that was used for benchmarking and the proposed feature screening method. Section 3 presents the performance of the novel screening method and compares it with other reduction algorithms. Finally, section 4 provides our conclusions and suggests future research directions.

## 2. Data and methodology

### 2.1. Synthetic dataset

To compare the performance of the proposed RFMS with a wide range of feature screening methods, an ultrahigh-dimensional, multiclass feature space—with ground truth and some additional side information on the usefulness of the features—was employed. This feature space imitates the key properties of the private signature dataset of Cursor Insight, which was the winner of the ICDAR competition on signature verification and writer identification in 2015 [29]. Moreover, it was compiled by using the *BiometricBlender* data generator [30]. The *BiometricBlender* Python package provides an alternative to real biometric datasets, which are typically not freely accessible and cannot be published for industrial reasons. The package is publicly available on GitHub [31].

The following parameters were set during the data generation process:

- `n-classes` = 100;
- `n-samples-per-class` = 64;
- `n-true-features` = 100;
- `n-fake-features` = 300;
- `min-usefulness` = 0.5;
- `max-usefulness` = 1;
- `location-sharing-extent` = 50;
- `location-ordering-extent` = 20;
- `n-features-out` = 10 000;
- `blending-mode` = 'logarithmic';
- `min-count` = 4;
- `max-count` = 8;
- `random-state` = 137.

The resulting dataset contains a categorical target variable with 100 unique classes and 10 000 intercorrelated features. The features are composed in two steps. First, true and fake features are drawn from a given distribution. These are called hidden features. The `{min/max}-usefulness` parameters regulate how much information a true feature reveals about the class identity. A shared location creates groups of classes such that the samples of any particular class are indistinguishable based on the feature values drawn at this location. The ordering makes true features correlated, so their information content becomes subadditive. Fake features contain random noise. In the second step, the output features are produced as a combination of a number of features, set by the `{min/max}-count` parameter. Logarithmic blending results in higher-order correlations. For an illustration, see figure 1. For details, please refer to the related paper [30].

### 2.2. RFMS

Before we describe the steps of the proposed screening algorithm, several notations have to be introduced. Let $y \in \{1, 2, \ldots, k\}$ be a categorical target variable that contains $k$ different classes ($k \in \mathbb{N}^+$, $k \geqslant 2$), and let $\mathbf{x} = \langle x_1, x_2, \ldots, x_n \rangle$ be the tuple of input features ($n \in \mathbb{N}^+$). (Note that the method may straightforwardly be applied to continuous target variables as well.) Moreover, let $\alpha, \beta \in \mathbb{N}^+$ be predefined parameters such that $1 \leqslant \beta \leqslant \alpha \leqslant n$, where $\alpha$ denotes the size of the subsets that the feature space will be divided into, and $\beta$ denotes the number of features that will be selected by the algorithm. For optimal values of $\alpha$ and $\beta$, see the `step-size` and `reduced-size` parameters in the appendix.

**Preparation.** First, the input features of $\mathbf{x}$ are arranged in random order. Formally, let $\pi$ be a random permutation of $\{1, 2, \ldots, n\}$, then

$$\mathbf{x}_\pi = \left\langle x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)} \right\rangle$$

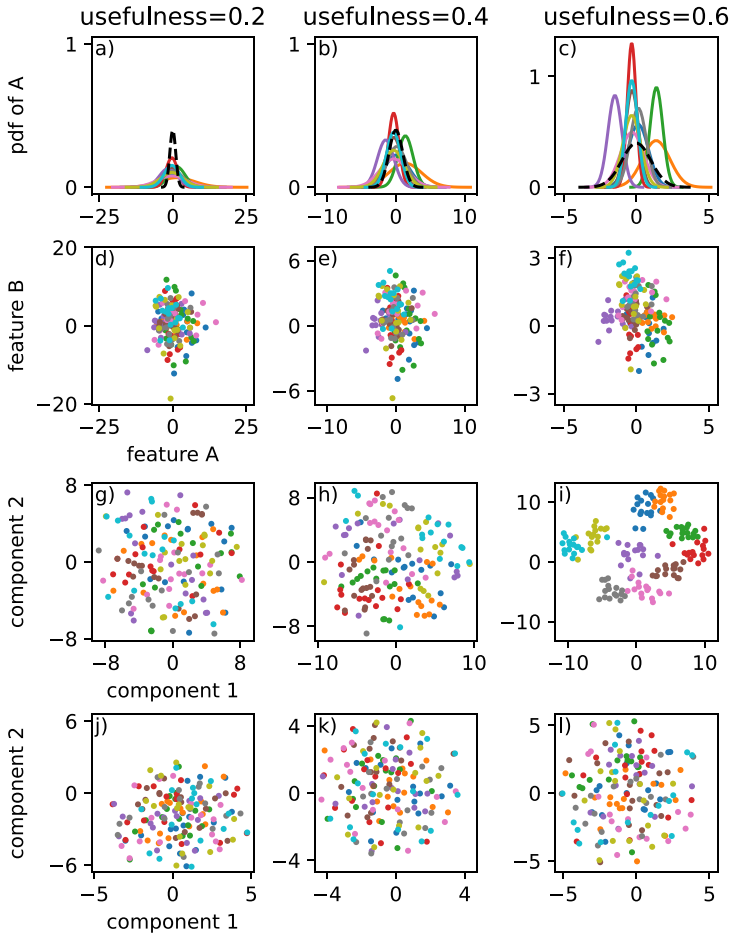


**Figure 1.** A depiction of the generated feature space. Columns indicate different values of the usefulness parameter: 0.2, 0.4, and 0.6, from left to right. (a)–(c) The distributions of a single true feature. (d)–(f) The shared distribution of the first two true features. True features carry independent information about the class identity. (g)–(i) The t-SNE representation of the feature space created by ten true features. (j)–(l) The t-SNE representation of the 100-dimensional output feature space obtained by logarithmically blending the true features, but excluding fake features. With the inclusion of fake features that contain only noise, samples from the same class get even more scattered in the feature space.

denotes the randomly ordered tuple of input features. $\mathbf{x}_\pi$ is then divided into $m = \lceil n/\alpha \rceil$ subsets as follows:

$$\begin{aligned}
\mathbf{x}_\pi^1 &= \left\langle x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(\alpha)} \right\rangle, \\
\mathbf{x}_\pi^2 &= \left\langle x_{\pi(\alpha+1)}, x_{\pi(\alpha+2)}, \ldots, x_{\pi(2\alpha)} \right\rangle, \\
&\vdots \\
\mathbf{x}_\pi^j &= \left\langle x_{\pi((j-1)\alpha+1)}, x_{\pi((j-1)\alpha+2)}, \ldots, x_{\pi(j\alpha)} \right\rangle \quad (1 \leqslant j < m), \\
&\vdots \\
\mathbf{x}_\pi^m &= \left\langle x_{\pi((m-1)\alpha+1)}, x_{\pi((m-1)\alpha+2)}, \ldots, x_{\pi(n)} \right\rangle.
\end{aligned}$$

**Iteration.** In this step, we iterate over the above mentioned subsets by selecting the $\beta$ most important features from a subset, adding them to the next subset, and repeating this process until the $\beta$ most important features are selected from the last subset. Formally, for $1 \leqslant i \leqslant m$, let

$$\bar{\mathbf{x}}_\pi^i = \mathbf{x}_\pi^i \frown \mathbf{z}^{i-1} = \left\langle \bar{x}_1^i, \bar{x}_2^i, \ldots, \bar{x}_t^i \right\rangle$$

(i.e. the concatenation of the two tuples), where $t = |\mathbf{x}_\pi^i| + \beta \leqslant \alpha + \beta$, $\mathbf{z}^0 = \langle\,\rangle$ is an empty tuple, and $\mathbf{z}^i$ $(1 \leqslant i < m)$ will be defined below. (Note that $\bar{\mathbf{x}}_\pi^1 = \mathbf{x}_\pi^1$.) The relative feature importance of $\bar{\mathbf{x}}_\pi^i$ on $y$ is identified by using RF classification. The importance of a feature is determined by the total number of times it appears in the classification forest (often termed the *selection frequency*).

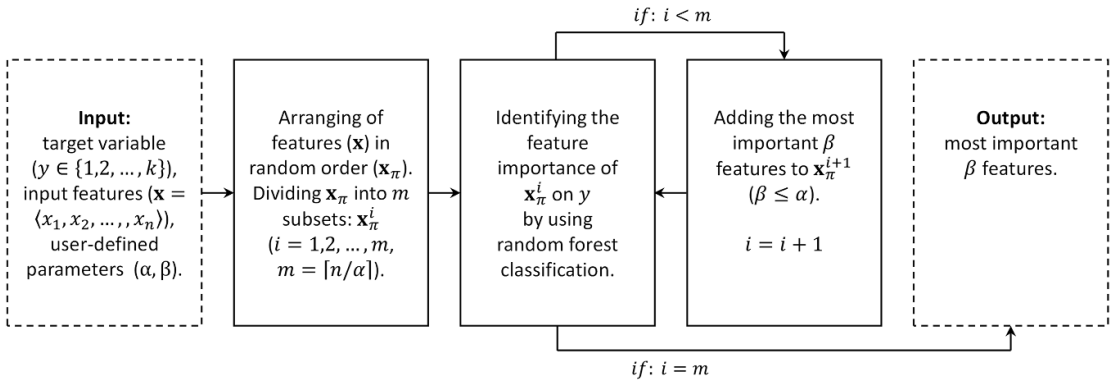


**Figure 2.** Steps of the RFMS.

The most important $\beta$ features of $\bar{\mathbf{x}}_\pi^i$ are stored in:

$$\mathbf{z}^i = \left\langle \bar{x}^i_{G_i(1)}, \bar{x}^i_{G_i(2)}, \ldots, \bar{x}^i_{G_i(\beta)} \right\rangle,$$

where $G_i : \{1,2,\ldots,\beta\} \to \{1,2,\ldots,t\}$ is an injective function that sorts the features in $\mathbf{z}^i$ in descending order of their importance.

**Result.** The $\beta$ features considered most important by the RFMS are found in:

$$\mathbf{z} = \mathbf{z}^m = \left\langle \bar{x}^m_{G_m(1)}, \bar{x}^m_{G_m(2)}, \ldots, \bar{x}^m_{G_m(\beta)} \right\rangle.$$

The aforementioned steps of the calculation are illustrated in figure 2 and described in algorithm 1.

**Algorithm 1.** The RFMS algorithm.

**Require:** RF: random forest classifier with set hyperparameters
**Require:** **X**: $(s, n)$-sized matrix ($s$: sample count, $n$: feature count) ▷ input features
**Require:** **y**: $s$-sized vector ▷ target variable
**Require:** $\alpha$: integer, st. $1 \leqslant \alpha \leqslant n$ ▷ number of input features per round
**Require:** $\beta$: integer, st. $1 \leqslant \beta \leqslant \alpha$ ▷ number of features to keep per round
$m \leftarrow \lceil n/\alpha \rceil$ ▷ number of tournament rounds
$\pi \leftarrow$ permutation($n$) ▷ take a random permutation
**z** $\leftarrow$ list() ▷ initialize top features as empty list
**for** $j = 1 \to m$ **do**
candidates $\leftarrow$ **z** $\cup\, \pi[(j-1)\alpha + 1, \ldots, \min(j\alpha, n)]$
importances $\leftarrow$ RF.importances(**X**[1 : $s$, candidates], **y**)
**z** $\leftarrow$ candidates[argsort(importances)].take($\beta$) ▷ keep best $\beta$
**return z**

## 3. Results and discussion

To compare the performance of the RFMS with off-the-shelf screening methods, we completed the following measurements:

(i) We measured the maximum accuracy of three basic classifiers—$k$-nearest neighbors ($k$NN) [32, 33], support vector classifier (SVC) [34], and RF [26]—on the full feature set by using $n$-fold cross-validation. The optimal parameters of the classifiers were identified via a grid search.
(ii) We performed screening by using four different methods (including our method), thus resulting in the requested number of screened features (from 10 to 500) per method. The tested screening methods included PCA [35, 36], FA [37, 38], $k$-best [39], and RFMS.
(iii) We measured the maximum accuracy of the three classifiers on each of the screened feature sets by using $n$-fold cross-validation.
(iv) For every step above, we also measured the CPU usage.

**Table 1.** Classification results on the 6400 × 10 000 dataset for three basic classifiers and various reduction algorithms. (a) Only the best accuracy among all of the parameters is reported. Bold values indicate the highest accuracies for each classifier. (b) Screening times are the CPU times of the feature screening step and correspond to the best accuracy shown above. (c) Fitting times are defined as the CPU times after the reduction step and correspond to the best accuracy shown above.

(a) Classification accuracy

| Reduction: | | None | PCA | FA | *k*-best | RFMS |
|---|---|---|---|---|---|---|
| Class.: | *k*NN | 0.043 | 0.092 | 0.101 | 0.313 | **0.381** |
| | SVC | 0.244 | 0.226 | 0.420 | 0.518 | **0.614** |
| | RF | 0.428 | 0.155 | **0.614** | 0.533 | 0.604 |

(b) Screening time

| Reduction: | | None | PCA | FA | *k*-best | RFMS |
|---|---|---|---|---|---|---|
| Class.: | *k*NN | — | 6.5 s | 21 s | 1.63 s | 11 464 s |
| | SVC | — | 54.6 s | 457 s | 1.48 s | 11 266 s |
| | RF | — | 25.8 s | 471 s | 1.48 s | 10 931 s |

(c) Fitting time

| Reduction: | | None | PCA | FA | *k*-best | RFMS |
|---|---|---|---|---|---|---|
| Class.: | *k*NN | 0.76 s | 0.0062 s | 0.0061 s | 0.010 s | 0.0082 s |
| | SVC | 185 s | 11.9 s | 11.9 s | 11.3 s | 11.6 s |
| | RF | 1145 s | 223 s | 233 s | 192 s | 59.6 s |

We did not benchmark our screening algorithm against artificial neural networks (ANNs) for various reasons:

- convolutional neural networks (CNNs) are more suited for tasks in which features obey a topology, e.g. image processing;
- autoencoders implement unsupervised learning and are incapable of finding features which facilitate the identification of any specific set of classes;
- ANNs in general are not well-suited for generating explainable models, which is often a requirement in the biometric domain;
- methods using ANNs are often legally restricted to prevent the restoration of original signatures.

Furthermore, we have considered several screening algorithms mentioned in section 1 for benchmarking, but we have been unable to find one that could be successfully evaluated on our artificial datasets for the following reasons:

- many of them work only on binary classification problems;
- most of them, according to the cited papers, have only been tested on datasets that are one to two magnitudes smaller;
- many of them did not have an off-the-shelf implementation available;
- those few that we could try took an unreasonably long time to finish, e.g. SIFE [25] took 2 h on 1/20th of the full feature set.

The highest classification accuracies for each combination, along with their screening and fitting times, are summarized in table 1. The optimized hyperparameters that were used during the application of the RFMS method can be found in the appendix.

Based on the results, the RFMS and FA methods outperformed both PCA and *k*-best screening in accuracy. The highest accuracy was achieved by using the RFMS–SVC and FA–RF pairs (61.4%); however, the latter combination required considerably lower screening time. Notably, depending on the persistence of the features (see, e.g. [40]), the screening was performed relatively infrequently in comparison with the fitting procedure, in which the combination comprising RFMS proved to be relatively fast. Furthermore, in exchange for a slower screening procedure, RFMS offers several advantages over the FA method. These advantages are detailed below.

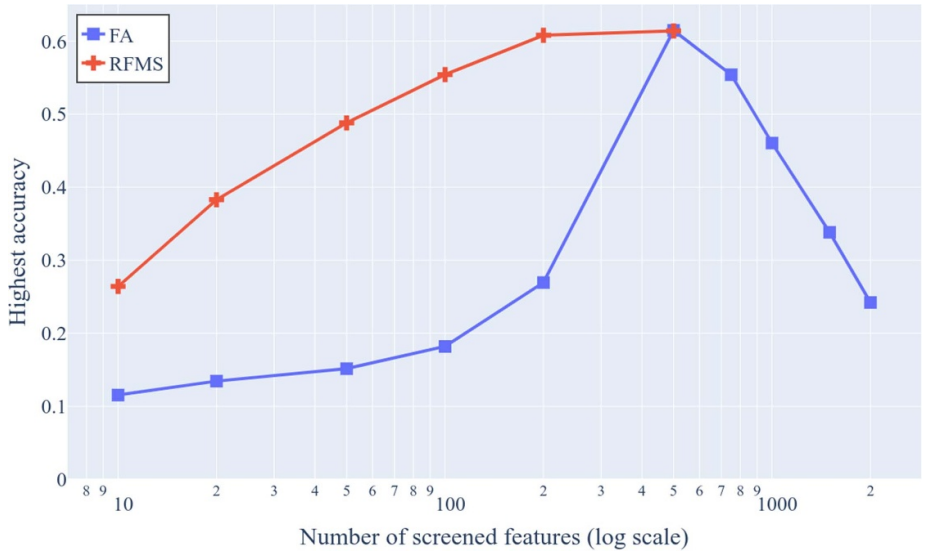


**Figure 3.** Convergence of highest accuracy as a function of the number of screened features (components). Accuracy is measured on a scale of 0–1. RFMS converges to the optimum much quicker than FA.

**Compatibility with numerical and categorical features, as well as missing values.** Since RFMS uses RFs to determine feature importances, and RFs are inherently insensitive to value domains and ranges and also tolerate missing values, RFMS itself is very forgiving regarding the feature values in the dataset.

**Potential cost reduction in feature computation.** To use FA on an incoming sample, its full feature set must be computed before the transformation can be applied. The trained model only works on the transformed feature set. In contrast, the output of RFMS is a transformation-free subset of the original feature set. This facilitates the interpretation of the resulting features; in addition, once RFMS has finished, and we have the set of optimal features, only these features need to be computed on any further incoming samples. This could be a significant factor in saving on cost and time in a production system.

**Suitability for several classifiers.** Although the combination of FA and RF resulted in a high accuracy and low screening time, the accuracy of the same FA output with SVC and *k*NN classifiers produced significantly weaker results (accuracy of 42% and 10%, respectively). However, for the RFMS output, SVC performed slightly better than RF (just as well as the FA–RF combination), and even the accuracy of the *k*NN classifier at 38.1% was much closer to the top performers.

**Robustness.** If we further investigate past the highest accuracies for every combination and observe how the accuracy changes with the adjustment of the hyperparameters, we can conclude that FA is quite sensitive. If we reduce the number of screened features (components) from 500 to 250, the highest achievable accuracy drops to 33.1%. A further reduction to 125 results in an accuracy of only 25%. A similar performance drop is observable if we begin to increase the number of features from 500. However, with RFMS, a reduction in the number of screened features from 500 to 200 only slightly reduces the best accuracy to 60.8%, and with a further reduction to 100, the accuracy is still 55.4%. We observed this behavior with high probability when the degrees of freedom of the data were well defined, but the FA was requested to produce fewer features.

Figure 3 summarizes both trends on a single plot, thus demonstrating how the highest achievable accuracy converges to its global optimum as the number of screened features increases. Note that the deviation from the plotted accuracy values with the randomization of the selection and measurement process is negligible.

In addition, by adjusting the RFMS hyperparameters, the screening time can be significantly reduced without compromising the classification accuracy. For example, with the right combination, the screening time can be decreased to 2143 s (merely 1/5th of the highest value in table 1), while the achievable accuracy is

still 60%. The fastest run in our test occurred for 1738 s (15% of the longest screening time), and even that output could achieve a 57.3% accuracy (93.4% of the overall highest accuracy).

**Performance on proprietary datasets.** We have extensively used RFMS on our own proprietary biometric feature sets; although we cannot publicly share these datasets, we can share our experiences. We found that the FA–RF pair typically performs worse than the combination of RFMS–RF for real feature sets. In one particular case, we trained both screening methods on a dataset of 10 000 classes, 81 000 samples, and 18 700 input features and targeted 200 output features. We subsequently measured the performance of the screened features by using a disjunct dataset of 44 classes and 58 000 samples (as well as the same number of features). The best classification accuracy that we could obtain on an FA transformed feature set was approximately 82%, while the RFMS-filtered output could elicit classification rates up to as high as 93%, albeit with the screening time being significantly longer (both values have been measured with five-fold cross-validation). However, given the sensitive and proprietary nature of the dataset, we cannot provide hard evidence for this claim.

Besides biometric authentication, we have also successfully applied RFMS as part of our ML toolchain in a pilot project aimed at predicting risk levels of certain financial transactions, such as personal loans, based on biometric features extracted from the applicants' signatures. Our predictions, based on our preliminary measurements, have been found to complement the risk level indicators applied by our partnering financial institute and increased the overall accuracy of the full risk analysis model by as much as 10% [41].

## 4. Conclusions and future work

Research on feature screening has grown rapidly in recent years; however, screening ultralarge, multiclass data is still in its infancy. To narrow this gap in the research, we presented a novel method known as RFMS that can be effectively applied in such circumstances. Due to the fact that ultrahigh-dimensional, multiclass data are typically encountered in biometrics, the RFMS was benchmarked on a synthetic feature space that imitates the key properties of a real (private) signature dataset. Based on the results, the RFMS is on par with industry-standard feature screening methods, and it also possesses many advantages over these methods due to its flexibility and robustness, as well as its transformation-free operation. The Julia package that implements RFMS is publicly available on GitHub [28].

The difference in maximum accuracy that was achieved on real and synthetic data suggests that the synthetic data generator used for tests does not yet reproduce all the properties of real data that challenge feature screeners, and this scenario is especially true for factor analysis. Therefore, it would be important to explore the properties of real data that cause this difference and to further develop *BiometricBlender* in this direction, which could subsequently enable more realistic tests. However, at the time of writing this paper, we were unable to find publicly available datasets that were suited for our purposes. The freely available databases, such as those published in the UCI ML repository [24], either had too few features, samples, or classes, or the nature of the features clearly destined them for use with other kinds of classifiers—e.g. the pixels of high-resolution images, for which CNNs, deep neural networks, etc are better candidates.

Note that our method can be straightforwardly generalized through the following modifications:

(i) Replace the RF and importance metrics with less common alternatives that may yield better performance.
(ii) Explore various forms of elimination tournaments, such as multiple passes through the input or alternative scoring methods like the Elo or Glicko [42] systems. This could potentially improve accuracy, particularly when information is intricately distributed across multiple 'entangled' features.
(iii) Reduce screening time by employing additional parallel computations (RF construction already leverages multiple threads when available).

To further develop the RFMS method, the following future works are suggested:

(i) Filter highly correlated variables in every iteration (e.g. [43]) just before classification, as this could improve the importance of the features that are proposed by the method.
(ii) Identify the means of automatically determining the number of important features to be retained per cycle, thus allowing for all the important features to be kept and most unnecessary features to be dropped. This could improve both accuracy and computation time.
(iii) Hyperparameter optimization is typically not viable with brute force due to lengthy computation times. Handy visualization tools could provide useful hints for manual boosting.

## Data availability statement

The applied feature space was compiled by using the *BiometricBlender* data generator [30], which is publicly available on GitHub [31].

## Acknowledgments

The authors would like to thank Erika Griechisch and Júlia Boróka Németh (Cursor Insight, London) and András Telcs (Wigner Research Centre for Physics, Budapest) for their valuable comments and advice. M S, M T K, and Z S thank the support of Hungarian Research Network (formerly Eötvös Loránd Research Network), Grant SA-114/2021 and on behalf of the project 'Identifying Hidden Common Causes: New Data Analysis Methods' for access to the HUN-REN Cloud (see [44]; https://science-cloud.hu/), which helped us to achieve the results that are published in this paper. Z S and M T K received support from the Hungarian Scientific Research Fund (OTKA/NRDI Office) under Contract Numbers K135837 and PD142593, respectively. M T K thanks the support of the Ministry of Innovation and Technology NRDI Office within the framework of the MILAB Artificial Intelligence National Laboratory Program.

## Author contributions statement

**Gergely Hanczár**: Conceptualization, Supervision, Methodology, Writing—Original Draft, Writing—Review & Editing, Project administration; **Marcell Stippinger**: Software, Methodology, Validation, Formal analysis, Investigation, Writing—Original Draft, Writing—Review & Editing; **Dávid Hanák**: Software, Methodology, Validation, Formal analysis, Investigation, Visualization, Writing—Original Draft, Writing—Review & Editing; **Marcell T Kurbucz**: Methodology, Investigation, Visualization, Writing—Original Draft, Writing—Review & Editing; **Olivér M Törteli**: Software, Validation, Formal analysis, Investigation, Data Curation; **Ágnes Chripkó**: Formal analysis, Investigation, Writing—Original Draft, Writing—Review & Editing; **Zoltán Somogyvári**: Supervision, Methodology, Validation, Writing—Original Draft, Writing—Review & Editing.

## Conflict of interest

We wish to make readers aware of the following facts that may be considered potential conflicts of interest, as well as make them aware of significant financial contributions to this work. The nature of the potential conflict of interest involves the fact that some authors work for Cursor Insight, which is an IT company targeting human motion analysis, person classification, and identification based on large-scale biometric data in particular.

## Appendix

RFMS was based on a Julia package that is publicly available on GitHub [28]. Its hyperparameters have been optimized via a grid search to identify a combination that produces the highest classification accuracy, as well as to observe the effect of changing the hyperparameters on the outcome.

In all cases, a fixed random seed of 20 230 125 was used to make the process deterministic. A fixed value of 0.7 was set for the `partial-sampling` parameter. Finally, 100 random features were added to the mix before screening as *canaries*. If any of these features had appeared in the final set of screened features on the output, we could have been confident that any less important features were simply noise. However, none of our total 3969 measurements (539 screening configurations combined with nine different classifiers, minus the contradicting combinations) stumbled upon a random feature among the screened ones; therefore, we were confident that the screening process identified truly relevant and meaningful features.

Table A1 summarizes the best four hyperparameter combinations, one for each of the three tested classifiers, plus one that produced the smallest screening time.

To visualize the learning curve of the screening process, we used the highest-performing scenario from table A1 and ran its SVC classifier on the output of each of the 20 rounds of screening. (After 20 rounds, each feature is encountered exactly once by the screener.) The highest-ranking random feature appeared at the 159th position after round 3, therefore we fed the 150 top features of each round to the SVC classifier. (In the winning scenario, we were able to use 500 screened features and reach an accuracy of 61.4%, because after the 20th round, the highest-ranking random feature was only 711th. When using only 150, accuracy tops out at 54%.) The accuracy values are depicted in figure A1. As expected, the accuracy of the classifier

**Table A1.** Optimal screening hyperparameters, the corresponding screening times, and the best achievable classification accuracies for various classifiers and fastest screening according to grid search results.

| Parameter | *k*NN | SVC | RF | Fastest |
|---|---|---|---|---|
| `reduced-size` | 200 | 500 | 200 | 200 |
| `step-size` | 505 | 505 | 505 | 2020 |
| `n-subfeatures` | 200 | 100 | 500 | 1000 |
| `n-trees` | 500 | 1000 | 200 | 100 |
| `min-samples-leaf` | 1 | 1 | 1 | 40 |
| `min-purity-increase` | 0.01 | 0.1 | 0.1 | 0 |
| Screening time: | 11 464 s | 10 931 s | 11 266 s | 1738 s |
| Accuracy: | | | | |
| • with *k*NN | 38.1% | | | 33.9% |
| • with SVC | | 61.4% | | 56.7% |
| • with RF | | | 60.4% | 57.3% |

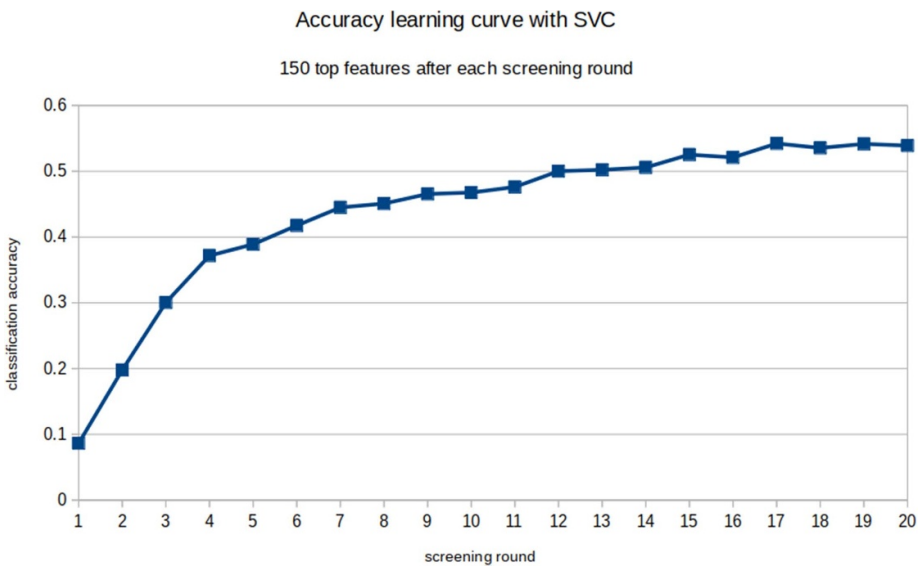


**Figure A1.** The learning curve of the SVC classifier after each RFMS screening round of the highest performing scenario.

increases with each round in a roughly logarithmic manner, flattening out as we get closer to the end of the screening procedure.

## ORCID iDs

Gergely Hanczár https://orcid.org/0000-0002-0222-1400
Marcell Stippinger https://orcid.org/0000-0002-9954-8089
Dávid Hanák https://orcid.org/0000-0003-0678-9885
Marcell T Kurbucz https://orcid.org/0000-0002-0121-6781
Olivér M Törteli https://orcid.org/0000-0002-2148-9189
Ágnes Chripkó https://orcid.org/0000-0002-2863-5257
Zoltán Somogyvári https://orcid.org/0000-0002-4385-3025

## References

[1] Wang H 2009 *J. Am. Stat. Assoc.* **104** 1512–24
[2] Tan H, Wang G, Wang W and Zhang Z 2022 *J. Appl. Stat.* **49** 411–26
[3] Li J and Liu H 2017 *IEEE Intell. Syst.* **32** 9–15
[4] Speiser J L, Miller M E, Tooze J and Ip E 2019 *Expert Syst. Appl.* **134** 93–101
[5] Jung S and Marron J S 2009 *Ann. Stat.* **37** 4104–30
[6] Kosztyán Z T, Kurbucz M T and Katona A I 2022 *Knowl.-Based Syst.* **251** 109180
[7] Yang B, Yin X and Zhang N 2019 *J. Multivariate Anal.* **173** 480–93
[8] Fan J and Lv J 2008 *J. R. Stat. Soc.* B **70** 849–911
[9] Fan J and Fan Y 2008 *Ann. Stat.* **36** 2605

[10] Mai Q and Zou H 2013 *Biometrika* **100** 229–34
[11] Lai P, Song F, Chen K and Liu Z 2017 *Stat. Probab. Lett.* **125** 141–8
[12] Roy S, Sarkar S, Dutta S and Ghosh A K 2022 arXiv:2205.03831
[13] Székely G J and Rizzo M L 2005 *J. Multivariate Anal.* **93** 58–80
[14] Baringhaus L and Franz C 2010 *Stat. Sin.* **20** 1333–61 (available at: www.jstor.org/stable/24309507)
[15] Mai Q and Zou H *et al* 2015 *Ann. Stat.* **43** 1471–97
[16] Ni L, Fang F and Wan F 2017 *Metrika* **80** 805–28
[17] Ni L and Fang F 2016 *J. Nonparametr. Stat.* **28** 515–30
[18] Amini F and Hu G 2021 *Expert Syst. Appl.* **166** 114072
[19] Dong H, Li T, Ding R and Sun J 2018 *Appl. Soft Comput.* **65** 33–46
[20] Gu S, Cheng R and Jin Y 2018 *Soft Comput.* **22** 811–22
[21] Zhang Y, Zhang J, Guo Y and Sun X 2016 *J. Intell. Fuzzy Syst.* **31** 2807–12
[22] Hosseini E S and Moattar M H 2019 *Appl. Soft Comput.* **82** 105581
[23] Xue Y, Tang T, Pang W and Liu A X 2020 *Appl. Soft Comput.* **88** 106031
[24] Kelly M, Longjohn R and Nottingham K 2023 The UCI machine learning repository (available at: https://archive.ics.uci.edu)
[25] Saadatmand H and Akbarzadeh-T M R 2023 *Appl. Soft Comput.* **142** 110240
[26] Breiman L 2001 *Mach. Learn.* **45** 5–32
[27] Wang G, Fu G and Corcoran C 2015 *BMC Genet.* **16** 1–11
[28] Hanczár G, Stippinger M, Hanák D, Kurbucz M T, Törteli O M, Chripkó A, Hergert L, Németh J and Somogyvári Z 2023 *FeatureScreening* (GitHub) (available at: https://github.com/cursorinsight/FeatureScreening.jl)
[29] Malik M I, Ahmed S, Marcelli A, Pal U, Blumenstein M, Alewijns L and Liwicki M 2015 ICDAR2015 competition on signature verification and writer identification for on- and off-line skilled forgeries (SigWIcomp2015) *2015 13th Int. Conf. on Document Analysis and Recognition (ICDAR)* (IEEE) pp 1186–90
[30] Stippinger M, Hanák D, Kurbucz M T, Hanczár G, Törteli O M and Somogyvári Z 2023 *SoftwareX* **22** 101366
[31] Stippinger M, Hanák D, Kurbucz M T, Hanczár G, Törteli O M, Hergert L and Somogyvári Z 2022 *BiometricBlender* (GitHub) (available at: https://github.com/cursorinsight/biometricblender)
[32] Fix E and Hodges J L 1989 *Int. Stat. Rev.* **57** 238–47
[33] Cover T and Hart P 1967 *IEEE Trans. Inf. Theory* **13** 21–27
[34] Vapnik V N 1998 *Statistical Learning Theory* (Wiley)
[35] Pearson K 1901 *London, Edinburgh Dublin Phil. Mag. J. Sci.* **2** 559–72
[36] Hotelling H 1933 *J. Educ. Psychol.* **24** 417
[37] Spearman C 1904 *Am. J. Psychol.* **15** 201–92
[38] Yong A G *et al* 2013 *Tutorials Quant. Methods Psychol.* **9** 79–94
[39] Wong K W, Tsui C Y, Cheng R K and Mow W H 2002 A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels *2002 IEEE Int. Symp. on Circuits and Systems (ISCAS)* vol 3 (IEEE) p III
[40] Friedman L, Nixon M S and Komogortsev O V 2017 *PLoS One* **12** e0178501
[41] Hanczár G, Törteli O M, Ovád N, Griechisch E, Hanák D, Hergert L, Papp L, Zelczer T and Golda B 2023 Ügyfelek fizetőképességének, megbízhatóságának becslése testbeszéd és más motoros koordináció alapján *Hungary Patent No.* P2300200 (Hungarian Intellectual Property Office) (available at: www.sztnh.gov.hu/en)
[42] Glickman M E 1995 *Boston Univ.* **16** 16–17
[43] Mitra P, Murthy C and Pal S 2002 *IEEE Trans. Pattern Anal. Mach. Intell.* **24** 301–12
[44] Héder M *et al* 2022 *Inf. Tarsadalom* **22** 128