Topological Feature Selection

Antonio Briola [1 2] **Tomaso Aste** [1 2 3]

Abstract

In this paper, we introduce a novel unsupervised, graph-based filter feature selection technique which exploits the power of topologically constrained network representations. We model dependency structures among features using a family of chordal graphs (i.e. the Triangulated Maximally Filtered Graph), and we maximise the likelihood of features' relevance by studying their relative position inside the network. Such an approach presents three aspects that are particularly satisfactory compared to its alternatives: (i) it is highly tunable and easily adaptable to the nature of input data; (ii) it is fully explainable, maintaining, at the same time, a remarkable level of simplicity; (iii) it is computationally cheap. We test our algorithm on 16 benchmark datasets from different application domains showing that it outperforms or matches the current state-of-the-art under heterogeneous evaluation conditions. The code and the data to reproduce all the results presented in the current research work are available at https://github.com/FinancialComputingUCL/Topological_Feature_Selection.

1. Introduction

In the era of big data, effective management of extensive feature spaces represents a genuine hurdle for scientists and practitioners. Only some features have significant relevance in real-world data, while the redundancy of the remaining ones actively inflates data-driven models' complexity. The search for always new and increasingly performing dimensionality reduction algorithms is hence justified by four primary needs: (i) reduce data maintenance costs (Wu et al., 2016; Eichelberger et al., 2017); (ii) reduce the impact of the 'curse of dimensionality' (Donoho et al., 2000; Poggio et al., 2017) on data-driven models; (iii) increase data-driven models' interpretability (Miao & Niu, 2016); (iv) reduce energy costs for models' training (Strubell et al., 2019; Patterson et al., 2021).

Dimensionality reduction addresses all these challenges by decreasing the complexity of the feature space while minimizing the information loss (Miner et al., 2012). Such a field can be further specified into two macro areas: (i) feature extraction (Mingqiang et al., 2008) and (ii) feature selection (Chandrashekar & Sahin, 2014). Feature extraction techniques originate new features by transforming the original ones into a space with a different dimensionality and then choosing linear or non-linear combinations of them. On the other hand, feature selection techniques directly choose a subset of features from the original set while maintaining their physical meaning. Usually, they represent a 'block' of a more complex pipeline including also a classification (or regression) step. The level of contamination among 'blocks' is more or less evident depending on the nature of the feature selection algorithm. Indeed, feature selection techniques are classified as (i) supervised (Huang, 2015); (ii) unsupervised (Solorio-Fernández et al., 2020): and (iii) semi-supervised (Sheikhpour et al., 2017). In the supervised case, features' relevance is usually assessed via their correlation degree with class labels or regression targets. These models take advantage of the learning performance of the classification (or regression) algorithm to refine the choice of the meaningful subset of features while maintaining, at the same time, 'block' independence from it. This loop interaction is convenient only in scenarios where retrieving labels and executing classification (or regression) tasks is computationally efficient. On the contrary, unsupervised feature selection is applied in scenarios where retrieving labels is costly. These algorithms select relevant features based on specific data properties (e.g. variance maximisation). Different classes of unsupervised feature selection approaches exist: (i) filters; (ii) wrappers; (iii) embedded methods. In filter-based methods, the feature selection stage is entirely independent from the classification (or regression) algorithm; in wrapper-based methods, the feature selection process takes advantage

[*]Equal contribution [1]Department of Computer Science, University College London, London, United Kingdom [2]Centre for Blockchain Technologies, University College London, London, United Kingdom [3]Systemic Risk Centre, London School of Economics, London, United Kingdom. Correspondence to: Antonio Briola <antonio.briola.20@ucl.ac.uk>.

Proceedings of the 2nd Annual Workshop on Topology, Algebra, and Geometry in Machine Learning (TAG-ML) at the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. 2023. *Copyright 2023 by the author(s).*

of classification (or regression) stage to improve its performance; in embedded methods, the feature selection step is embedded into the classification (or regression) algorithm in a way that the two 'blocks' take advantage from each other. Lastly, semi-supervised feature selection represents a hybrid approach with the highest potential applicability in real-world problems. Indeed, labels are often only partially provided, and semi-supervised learning techniques are specifically designed to learn from a reduced number of labelled data, efficiently handling, at the same time, a large number of unlabeled samples.

In its general formulation, given a starting set of features $F = \{f_1, \ldots, f_n\}$, where $n \gg 1$ is the cardinality, dimensionality reduction is an *NP-hard* problem (Guyon et al., 2008; Roffo et al., 2015) where the goal is selecting the optimal subset of features with cardinality $m \ll n$, among the $\binom{n}{m}$ possible combinations. Due to the exponentially increased time required to find the globally optimal solution, existing feature selection algorithms employ heuristic rules to find dataset-dependent sub-optimal solutions (Ge et al., 2016). In this paper, we introduce 'Topological Feature Selection' (TFS), a novel unsupervised, graph-based filter algorithm for feature selection which (i) builds a topologically constrained network representation of raw features' dependency structure and (ii) exploits their relative position into the graph to reduce the input's dimensionality while maximising the likelihood of features' relevance and minimising the information loss.

To prove the effectiveness of the proposed methodology, we test it against what is currently considered the state-of-the-art counterpart in unsupervised, graph-based filter feature selection approaches, i.e. Infinite Feature Selection (Inf-FS$_U$) (Roffo et al., 2020). The two feature selection algorithms are tested on 16 benchmark datasets from different application domains. The proposed training/test pipeline and the statistical validation stage are designed to handle datasets' imbalance and evaluate results based on fair performance metrics. The results are clear-cut. In most cases, TFS outperforms or equalises its alternative, redefining or teeing state-of-the-art performances. Our contribution to the existing literature is relevant since we propose an extraordinarily flexible, computationally cheap and remarkably intuitive (compared to its alternative) unsupervised, graph-based filter algorithm for feature selection, guaranteeing complete control of the dimensionality reduction process by considering only the relative position of features inside well-known graph structures.

The rest of the paper is organised as follows. In Section 2.1, we describe the datasets used to prove the effectiveness of the TFS algorithm. In Section 2.2, we review the theoretical foundations of TFS. In Section 2.3, we describe in a detailed manner the TFS methodology. In Section 2.5, we describe the experimental protocols, while obtained results are presented in Section 3. Finally, in Section 4, we discuss the meaning of our results and future research lines in this area.

2. Methods

2.1. Data

We prove TFS' effectiveness by running a battery of tests on 16 benchmark datasets (all in a tabular format) belonging to 7 different application domains (i.e. text data, face images data, hand written images data, biological data, digits data, spoken letters data and artificial data) (see Appendix A for further details). The data pre-processing pipeline consists of 3 different steps: (i) data reading and format unification; (ii) training/validation/test splitting; (iii) constant features pruning. The first step allows to read data coming from different sources and unify the format. The second step organises each dataset into a training, validation and a test set. In the third step, non-informative, constant covariates are detected on the training set and permanently removed from the training, validation and the test set. In Appendix B, we report datasets' specifics after the pre-processing step. We remark that most of the considered datasets are not affected by the constant features filtering step. Datasets are roughly balanced and the raw labels' distribution is preserved through the execution of stratification (Zeng & Martinez, 2000) during the training/validation/test splitting stage.

2.2. Information Filtering Networks

Information Filtering Networks (IFNs) (Mantegna, 1999; Aste et al., 2005; Tumminello et al., 2005; Barfuss et al., 2016; Massara et al., 2017) are an effective tool to represent and model dependency structures among variables characterising complex systems. In the past two decades, they have been extensively used in finance (Procacci & Aste, 2022; Briola et al., 2022; Wang & Aste, 2022a; Seabrook et al., 2022; Wang & Aste, 2022b; Vidal-Tomás et al., 2023), psychology (Christensen et al., 2018; Christensen, 2018), medicine (Hutter et al., 2019; Danoff et al., 2021) and biology (Song et al., 2008; 2012). Sometimes IFNs are also referred to as Correlation Networks (CNs). Such an association is, however, inaccurate. Indeed, the two methodologies slightly differ, with CNs being normally obtained by imposing a threshold that retains only the largest correlations among variables of the system, while IFNs being constructed imposing additional topological constraints (e.g. being a tree or a planar graph) and optimising specific global properties (e.g. likelihood) (Aste, 2022). Both methodologies end in the determination of a sparse adjacency matrix, $\boldsymbol{A}$, representing relations among variables in the system with the fundamental difference that the former approach gener-

ates a disconnected graph, while the latter guarantees the connectedness. Based on the nature of the relationships to be modelled (i.e. linear, non-linear), one can choose different metrics to build the adjacency matrix $\boldsymbol{A}$. In most cases, $\boldsymbol{A}$ is built on an arbitrary similarity matrix $\hat{\mathbf{C}}$, which often corresponds to a correlation matrix. From a network science perspective, $\hat{\mathbf{C}}$ can be considered as a fully connected graph where each variable of the system is represented as a node, and each pair of variables is joined by a weighted and undirected edge representing their similarity. Historically, the main IFNs were the Minimum Spanning Tree (MST) (Papadimitriou & Steiglitz, 1998; Mantegna, 1999) and the Planar Maximally Filtered Graph (PMFG) (Tumminello et al., 2005). MSTs are a class of networks connecting all the vertices without forming cycles (i.e. closed paths of at least three nodes) while retaining the network's representation as simple as possible (i.e. representing only relevant relations among variables characterising the system under analysis) (Briola & Aste, 2022). Prim's algorithm for MST construction sorts all edges' weights (i.e. similarities) in descending order and adds the largest possible edge weight among two nodes in an iterative way. The resulting network has $n-1$ edges and retains only the most significant connections, assuring, at the same time, that connectedness' property is fulfilled (Christensen et al., 2018). Despite being a powerful method to capture meaningful relationships in network structures describing complex systems, MST presents some aspects that can be unsatisfactory. Paradoxically, the main limit is represented by its tree structure (i.e. it cannot contain cycles) which does not allow to represent direct relationships among more than two variables showing strong similarity. The introduction of the Planar Maximally Filtered Graph (PMFG) (Tumminello et al., 2005) overcomes such a shortcoming. Similarly to MST, also the PMFG algorithm sorts edge weights in descending order, incrementally adding the largest ones while imposing planarity (Christensen et al., 2018). A graph is planar if it can be embedded in a sphere without edges crossing. Thanks to this, the same powerful filtering properties of the MST are maintained, and, at the same time, extra links, cycles and cliques (i.e. complete subgraphs) are added in a controlled manner. The resulting network has $3n-6$ edges and is composed of three- and four-nodes cliques. A nested hierarchy emerges from these cliques (Song et al., 2011): dimensionality is reduced in a deterministic manner while local information and the global hierarchical structure of the original network are retained. The PMFG represents a substantial step forward compared to the MST. However, it still presents two limits: (i) it is computationally costly and (ii) it is a non-chordal graph. A graph is said to be chordal if all cycles made of four or more vertices have a chord, reducing the cycle to a set of triangles. A chord is defined as an edge that is not part of the cycle but connects two vertices of the cycle itself. The advantage of chordal graphs is that they fulfill the independence assumptions of Markov (i.e., bidirectional or undirected relations) and Bayesian (i.e., directional relations) networks (Koller & Friedman, 2009; Christensen et al., 2018). The Triangulated Maximally Filtered Graph (TMFG) (Massara et al., 2017) has been explicitly designed to be a chordal graph while retaining the strengths of PMFG. The building process of TMFG (see Appendix C for further details) is based on a simple topological move that preserves planarity: it adds one node to the centre of three-nodes cliques by using a score function that maximises the sum of the weights of the three edges connecting the existing vertices. This addition transforms three-nodes cliques (i.e. triangles) into four-nodes cliques (i.e. tetrahedrons) characterised by a chord that forms two triangles and generates a chordal network (Christensen et al., 2018). Also in this case, the resulting network has $3n-6$ edges and is composed of three- and four-nodes cliques. TMFG has two main advantages compared to PMFG: (i) it can be used to generate sparse probabilistic models as a form of topological regularization (Aste, 2022) and (ii) it is computationally more efficient. On the other hand, the two main limitations of chordal networks are that (i) they may add unnecessary edges to satisfy the property of chordality and (ii) their building cost can vary based on the chosen optimization function.

2.3. Topological Feature Selection

Topological Feature Selection (TFS) algorithm is a graph-based filter method to perform feature selection in an unsupervised manner. Given a set of features $F = \{f_1, \dots, f_n\}$, where $n \gg 1$ is the cardinality, we build the adjacency matrix $\boldsymbol{A}$ of the corresponding TMFG based on one of the following three metrics: (i) the Pearson's estimator of the correlation coefficient, (ii) the Spearman's rank correlation coefficient and (iii) the Energy coefficient (i.e. the weighted combination of two pairwise measures described later in this Section). Depending on the metric's formulation, it is possible to capture different kinds of interactions among covariates (e.g. linear or non-linear interactions).

The Pearson's estimator of the correlation coefficient for the two covariates f_i and f_j, is defined as:

$$r_{f_i,f_j} = \frac{\sum_{s=1}^{S}(f_{i,s} - \hat{\mu}_{f_i})(f_{j,s} - \hat{\mu}_{f_j})}{\hat{\sigma}_{f_i}\hat{\sigma}_{f_j}} \tag{1}$$

where S is the sample size, $f_{i,s}$ and $f_{j,s}$ are two sample points indexed with s, $\hat{\mu}$ is the sample mean and $\hat{\sigma}$ is the sample standard deviation. By definition, r_{f_i,f_j} has values between -1 (meaning that the two features are completely, linearly anti-correlated), and $+1$ (meaning that the two features are completely, linearly correlated). When $r_{f_i,f_j} = 0$, the two covariates are said to be uncorrelated. The Person's

estimator of the correlation coefficient heavily depends on the distribution of the underlying data and may be influenced by outliers. In addition to this, it only captures linear dependency among variables, restricting its applicability to real-world problems where non-linear interactions are often relevant (Shirokikh et al., 2013).

The Spearman's rank correlation coefficient is based on the concept of 'variables ranking'. Ranking a variable means mapping its realizations to an integer number that describes their positions in an ordered set. Considering a variable with cardinality $|s|$, this means assigning 1 to the realization with the highest value and $|s|$ to the realization with the lowest value.

The Spearman's rank correlation coefficient for the two covariates f_i and f_j, is defined as the Pearson's correlation between the ranks of the variables:

$$r_{s_{f_i,f_j}} = \frac{\sum_{s=1}^{S}(R_{f_{i,s}} - \hat{\mu}_{R_{f_i}})(R_{f_{j,s}} - \hat{\mu}_{R_{f_j}})}{\hat{\sigma}_{R_{f_i}}\hat{\sigma}_{R_{f_j}}} \quad (2)$$

where S is the sample size, $R_{f_{i,s}}$ and $R_{f_{j,s}}$ are the ranks of the two sample points indexed with s, $\hat{\mu}$ is the sample mean for R_{f_i} and R_{f_j} and $\hat{\sigma}$ is the sample standard deviation for R_{f_i} and R_{f_j}. If there are no repeated data samples, a perfect Spearman's rank correlation (i.e. $r_{s_{f_i,f_j}} = 1$ or $r_{s_{f_i,f_j}} = -1$ occurs when each of the features is a perfect monotone function of the other). Spearman's rank correlation technique is distribution-free and allows to capture monotonic, but not necessarily linear, relationships among variables (Shirokikh et al., 2013).

The Energy coefficient is a metric introduced by (Roffo et al., 2015), and it is used as the primary benchmark for comparison between our method and the current state-of-the-art. It is a weighted combination of two different pairwise measures defined as follows:

$$\phi_{f_i,f_j} = \alpha E_{f_i,f_j} + (1-\alpha)\rho_{f_i,f_j} \quad (3)$$

where $E_{f_i,f_j} = \max(\hat{\sigma}_{f_i}, \hat{\sigma}_{f_j})$, with $\hat{\sigma}$ being the sample standard deviation computed on features f_i and f_j normalized to the range $[0,1]$, $\rho_{f_i,f_j} = 1 - |r_{s_{f_i,f_j}}|$ and α is a threshold value with a value $\in [0,1]$. $\phi_{f_i,f_j} \in [0,1]$ analyses two features distributions (i.e. f_i and f_j) taking into account both their maximal dispersion (i.e. standard deviation) and their level of uncorrelation. Computing ϕ_{f_i,f_j} for all the features in F in a pairwise manner, one can define a matrix which is $n \times n$ symmetric and completely characterized by $n(n-1)/2$ coefficients. For simplicity, we refer to this matrix as $\hat{\mathbf{C}}$ too.

Once one of the above mentioned metrics is chosen and $\hat{\mathbf{C}}$ is computed, TFS applies the standard TMFG algorithm defined in Appendix C on the corresponding fully connected graph, creating a sparse chordal network which is able (i) to retain useful relationships among features, (ii) prune the weakest ones, and (iii) express a significant level of information flow among input variables.

The last step toward the selection of the most relevant features, is represented by the choice of the right nodes inside the TMFG. In this sense, multiple approaches of increasing complexity can be formulated. In this paper, which is a foundational one, we study the relative position of the nodes in the network computing their degree centrality. Degree centrality is the simplest and least computationally intensive measure of centrality. Typically, all the other centrality measures are strictly related (Lee, 2006; Valente et al., 2008). Given the sparse adjacency matrix $\boldsymbol{A}$ representing the TMFG, degree centrality of a node v is denoted as $\deg(v)$ and represents the number of neighbours (i.e. how many edges a node has) of v as follows:

$$\deg(v) = \sum_{w=1}^{n} A_{f_v,f_w} \quad (4)$$

where n is the cardinality of F and f_v and f_w are two features $\in F$. Despite its simplicity, degree centrality can be very illuminating and can be considered a crude measure of whether a node is influential or not in the TMFG (i.e. variables mostly contributing to the system's information flow). Once obtained $\deg(v)$ $\forall v \in$ TMFG, we rank these values in a descending order and we take the top k central nodes, where k is the cardinality of the features' subset we want to consider.

2.4. Benchmark method: Infinite Feature Selection (Inf-FS$_U$)

To prove the effectiveness of the proposed methodology, we test the TFS algorithm against the current state-of-the-art counterpart in unsupervised, graph-based filter feature selection techniques, i.e. Infinite Feature Selection (Inf-FS$_U$) (Roffo et al., 2020). [1] Inf-FS$_U$ represents features as nodes of a graph and relationships among them as weighted edges. Weights are computed as per in Equation 3. Each path of a given length over the network is seen as a potential set of relevant features. Therefore, varying paths and letting them tend to an infinite number permits the investigation of the importance of each possible subset of features. Based on this, assigning a score to each feature and ranking them in descendant order allows us to perform feature selection effectively. It is worth noting that Inf-FS$_U$ has a computational complexity equal to $\approx \mathcal{O}(n^3)$. In contrast, TFS has a

[1] The Python implementation of Inf-FS$_U$ algorithm used in this paper can be reached at https://github.com/fullyz/Infinite-Feature-Selection.

computational complexity equal to $\approx \mathcal{O}(n^2)$, with n being the number of the features.

2.5. Experiments

Once defined the training, validation and test set for each benchmark dataset, model hyper-parameters (see Table 1) are optimised adopting a parallel grid search approach. For Inf-FS$_U$, the first hyper-parameter to be optimised is α, which can take values between 0 and 1. To tune this parameter, we use a range of equally spaced realisations between 0.1 and 1.0, all at a distance of 0.1. The second hyper-parameter to be optimised is θ and represents a regularisation factor which, in the original paper (Roffo et al., 2015), has a fixed value equal to 0.9. Here, instead, we tune this parameter in the same way as α. In the case of TFS, the first hyper-parameter to be optimised is the metric used in the building process of the initial fully connected graph. As reported in Section 2.3, we test three different metrics: (i) the Pearson's estimator of the correlation coefficient, (ii) the Spearman's rank correlation coefficient and (iii) the Energy coefficient. The second hyper-parameter to be tuned is a boolean value which regulates the chance to use the coefficients mentioned above in a squared form. It is worth mentioning that, if the Energy metric is chosen, the corresponding $\hat{\mathbf{C}}$ is never squared since it already contains only positive values. The last hyper-parameter to be optimised is α, and it should be considered only when the Energy coefficient is chosen as a metric. The meaning of this hyper-parameter is the same as its homologous in the Inf-FS$_U$ model. All the models are finally evaluated on feature subsets with cardinalities $\in [10, 50, 100, 150, 200]$.

Table 1. Model dependent hyper-parameters search spaces. In the case of the TFS algorithm, the † symbol indicates that, if the Energy metric is chosen, the corresponding $\hat{\mathbf{C}}$ is never squared ($\hat{\mathbf{C}}$ already contains only positive values). The ‡ symbol, on the other hand, indicates that α parameter should be considered only when the Energy coefficient is chosen as a metric.

Model	Hyper-parameters
Inf-FS$_U$	α: [0.1: 0.1: 1.0] θ: [0.1: 0.1: 1.0]
TFS	metric: [Pearson, Spearman, Energy] square†: [True, False] $\alpha^\ddagger$: [0.1: 0.1: 1.0]

For each hyper-parameter combination, a stratified k-fold cross-validation with $k = 3$ is performed on the training set. The value of k is chosen to take into account labels' distributions reported in Appendix B. Results' reproducibility and a fair comparison between models are guaranteed by fixing the random seed for each step of the training/validation/test pipeline.

The meaningfulness of each subset of features chosen by the two algorithms is evaluated based on the classification performance achieved by three classification algorithms: (i) Linear Support Vector Classifier (LinearSVM); (ii) k-Nearest Neighbors Classifier (KNN); (iii) Decision Tree Classifier (Decision Tree). LinearSVM is a sparse kernel-based method designed to convert non-linearly separable problems in the low-dimensional space, into linearly separable problems in the higher-dimensional space, thereby achieving classification (Han et al., 2022). KNN is a lazy learning algorithm, which classifies test instances evaluating their distance from the nearest k training samples stored in an n-dimensional space (where n is the number of dataset's covariates) (Han et al., 2022). Finally, Decision Tree is a flowchart-like tree structure computed on training instances which classifies test samples tracing a path from the root to a leaf node holding the class prediction (Han et al., 2022). The inherently different nature of the three classifiers prevents from obtaining biased results for the two feature selection approaches. More details about chosen classifiers and their implementations are reported in Appendix D. Learning performances are evaluated based on three different metrics: (i) the Balanced Accuracy score (BA) (Mosley, 2013; Kelleher et al., 2020); (ii) the F1 score (F1); (iii) the Matthews Correlation Coefficient (MCC) (Baldi et al., 2000; Gorodkin, 2004; Jurman et al., 2012). We use the BA score for the hyper-parameters optimization process and as the reference metric to present results in Section 3. The BA score for the multi-class case is defined as:

$$BA = \frac{1}{|Z|}\left(\sum_{z \in Z} \frac{\text{TP}_z}{\text{TP}_z + \text{FN}_z} + \frac{\text{TN}_z}{\text{TN}_z + \text{FP}_z}\right). \quad (5)$$

TP is the number of outcomes where the model correctly classifies a sample as belonging to a positive class, when in fact it does belong to that class. TN is the number of outcomes where the model correctly classifies a sample as belonging to a negative class, when in fact it does not belong to that class. FP is the number of outcomes where the model incorrectly classifies a sample as belonging to a positive class, when in fact it does not belong to that class. FN is the number of outcomes where the model incorrectly classifies a sample as belonging to a negative class, when in fact it belongs to a positive class. $|Z|$ indicates the cardinality of the set of different classes.

General formulations for the F1 score, and the MCC are reported in Appendix F together with an extended version of results described later in this Section.

For each model and for each subset's cardinality, the hyper-parameters configuration which maximises the BA score while minimising the number of parameters is applied to test datasets. In order to assess the robustness of our findings,

we test if the results achieved by the two feature selection approaches are statistically different. To achieve this goal we use an improved version of the classic 5×2 cv paired *t*-test (Dietterich, 1998). The test is constructed as follows. Given two classifiers A and B and a dataset D, D is first randomly split into two balanced subsets D_1, D_2 (one for training and one for test). Both A and B are then estimated on D_1 and evaluated on D_2 obtaining performance measures a_1, b_1. The roles of the datasets are then switched by estimating A and B on D_2 and evaluating on D_1 which results in further performance measures a_2, b_2. The random division of D is performed for a total of 5 times, obtaining the matched performance evaluations $\{a_1, b_1\}, \{a_2, b_2\} \ldots, \{a_{10}, b_{10}\}$. The test statistic *t* is then computed as follows:

$$t = \frac{\sqrt{10}\bar{d}}{\hat{\sigma}}, \tag{6}$$

where $d_h = a_h - b_h$ for $h = 1, \ldots, 10$, is the difference between the matched performances metrics of the two classifiers, $\bar{d} = \frac{1}{10}\sum_{h=1}^{10} d_h$ and $\hat{\sigma}^2 = \frac{1}{10}\sum_{h=1}^{10}(d_h - \bar{d})^2$. t follows a *t*-distribution with 9 degrees of freedom and the null hypothesis is that *the two classifiers A and B are not statistically different in their performances*. Starting from this basic formulation of the 5×2 cv paired *t*-test, we simply increased the number of iterations, making it a 15×2 cv paired *t*-test, in order to increase the statistical robustness of the achieved results. Also in this case, results reproducibility is guaranteed through a strict control of random seeds.

3. Results

For both Inf-FS$_U$ and TFS, for each one of the three considered classifiers and for each feature subset cardinality $\in [10, 50, 100, 150, 200]$, results obtained running the hyper-parameter optimisation pipeline (see Section 2.5) are reported in Appendix E.

Table 2 reports out-of-sample Balanced Accuracy scores obtained using subset's cardinality-dependent optimal hyper-parameter configurations for LinearSVM, KNN and Decision Tree classifier respectively. For each dataset, we highlight in bold the best achieved result. If one classifier performs equally across multiple subsets' cardinalities, the winning configuration is the one which minimises the subset's cardinality itself. If one classifier performs equally under the two feature selection schema, the winning feature selection approach is the one which minimises the computational complexity (i.e. TFS).

To compare Inf-FS$_U$ and TFS, we consider three different measures: (i) the number of times a classifier achieves optimal results under each feature selection schema; (ii) the cross-datasets average balanced accuracy score; (iii) the cross-datasets average maximum drawdown ratio (i.e. the difference between the highest and the lowest achieved result).

TFS combined with LinearSVM classifier produces higher Balanced Accuracy scores in 14 out of 16 datasets (i.e. 87.5% of cases), while Inf-FS$_U$ combined with the same classifier has a higher Balanced Accuracy scores only in 2 out of 16 datasets (i.e. 12.5% of cases). Considering only the scenarios where TFS is the winning feature selection schema, we notice that in 1 case, the optimal cardinality is equal to 10, in 2 cases, the optimal cardinality is equal to 50 and 100, in three cases the optimal cardinality is equal to 150, and in 6 cases the optimal cardinality is equal to 200. When the Inf-FS$_U$ is used as feature selection algorithm and LinearSVM as classifier, the cross-datasets average Balanced Accuracy score grows from a value equal to 0.49 at cardinality 10 to a value of 0.75 at cardinality 200 with an increase of 26%. When the TFS is used as feature selection algorithm and LinearSVM as classifier, cross-datasets average Balanced Accuracy score grows from a value equal to 0.52 at cardinality 10 to a value of 0.78 at cardinality 200 with an increase of 26%. Finally, we notice that when the Inf-FS$_U$ is used as feature selection algorithm and LinearSVM as classifier, the average maximum drawdown ratio is equal to 31%. Using TFS as feature selection algorithm, instead, the average maximum drawdown ratio is equal to 28%.

TFS combined with KNN classifier produces higher Balanced Accuracy scores in 10 out of 16 datasets (i.e. 62.5% of cases), while Inf-FS$_U$ combined with the same classifier has higher Balanced Accuracy scores in 6 out of 16 datasets (i.e. 37.5% of cases). Considering only the scenarios where TFS is the winning feature selection schema, we notice that in 1 case, the optimal cardinality is equal to 10, 150 and 200, in 5 cases, the optimal cardinality is equal to 50, in two cases the optimal cardinality is equal to 100. When the Inf-FS$_U$ is used as feature selection algorithm and KNN as classifier, the cross-datasets average Balanced Accuracy score grows from a value equal to 0.46 at cardinality 10 to a value of 0.69 at cardinality 200 with an increase of 23%. When TFS is used as feature selection algorithm and KNN as classifier, the cross-datasets average Balanced Accuracy score grows from a value equal to 0.55 at cardinality 10 to a value of 0.71 at cardinality 200 with an increase of 16%. Finally, we notice that when Inf-FS$_U$ is used as feature selection algorithm and KNN as classifier, the average maximum drawdown ratio is equal to 23%. Using TFS as feature selection algorithm, instead, the average maximum drawdown ratio is equal to 21%.

TFS combined with Decision Tree classifier produces higher Balanced Accuracy scores in 13 out of 16 datasets (i.e. 81.25% of cases), while Inf-FS$_U$ combined with the same classifier produces higher Balanced Accuracy scores in 3

Table 2. Subset size-dependent, out-of-sample balanced accuracy scores using a LinearSVM, KNN and Decision Tree classifiers. For each dataset, we boldly highlight the combination between feature selection schema and classifier producing the best out-of-sample result. For each subset size, we report, in the last row, the number of times a feature selection approach outperforms the other across datasets.

	LinearSVM									
	10		**50**		**100**		**150**		**200**	
	Inf-FS$_U$	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**
PCMAC	0.52	0.50	0.57	0.67	0.59	0.70	0.61	**0.71**	0.62	0.69
RELATHE	0.47	0.49	0.43	**0.53**	0.51	0.53	0.44	0.49	0.53	0.53
COIL20	0.52	0.63	0.77	0.90	0.84	0.92	0.90	0.94	0.94	**0.96**
ORL	0.40	0.44	0.63	0.88	0.72	0.89	0.86	0.93	0.84	**0.94**
warpAR10P	0.33	0.44	0.56	0.78	0.72	0.85	0.70	**0.95**	0.75	0.85
warpPIE10P	0.85	0.89	0.95	**1.00**	0.98	1.00	1.00	1.00	1.00	1.00
Yale	0.14	0.33	0.25	0.50	0.39	0.67	0.37	0.69	0.53	**0.70**
USPS	0.72	0.65	0.90	0.90	0.91	0.92	0.92	**0.93**	0.92	0.93
colon	0.70	0.69	0.69	0.66	**0.92**	0.82	0.85	0.74	0.85	0.88
GLIOMA	**0.61**	0.25	0.30	0.30	0.30	0.38	0.60	0.41	0.59	0.25
lung	0.39	0.47	0.67	0.89	0.81	**0.95**	0.71	0.87	0.90	0.81
lung_small	0.49	0.57	0.76	0.79	0.82	0.68	0.79	0.75	0.82	**0.93**
lymphoma	0.22	0.50	0.58	0.96	0.78	0.87	0.90	0.82	0.81	**0.98**
GISETTE	0.50	0.49	0.48	0.47	0.51	**0.52**	0.47	0.50	0.49	0.50
Isolet	0.32	0.51	0.74	0.78	0.81	0.82	0.88	0.83	0.89	**0.89**
MADELON	0.59	**0.59**	0.58	0.56	0.55	0.57	0.54	0.57	0.57	0.57
# bests	5	11	3	13	2	14	5	11	2	14

	KNN									
	10		**50**		**100**		**150**		**200**	
	Inf-FS$_U$	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**
PCMAC	0.52	0.53	0.57	0.61	0.61	0.62	0.59	**0.63**	0.60	0.62
RELATHE	0.46	0.46	0.50	**0.57**	0.48	0.49	0.46	0.45	0.48	0.48
COIL20	0.70	0.82	0.86	0.93	0.93	0.93	0.96	0.94	**0.97**	0.93
ORL	0.38	0.52	0.52	**0.77**	0.62	0.70	0.73	0.71	0.72	0.77
warpAR10P	0.36	0.30	0.36	**0.51**	0.43	0.46	0.32	0.38	0.42	0.48
warpPIE10P	0.83	0.72	0.86	0.91	0.92	**0.97**	0.89	0.92	0.89	0.95
Yale	0.14	0.42	0.28	0.41	0.26	0.42	0.43	0.38	0.35	**0.49**
USPS	0.78	0.77	0.94	0.94	**0.96**	0.95	0.96	0.95	0.95	0.95
colon	0.77	0.82	0.89	0.77	0.89	0.85	**1.00**	0.70	0.85	0.77
GLIOMA	0.24	0.10	0.40	0.40	0.42	**0.62**	0.52	0.62	0.52	0.62
lung	0.33	0.51	0.65	**0.79**	0.71	0.65	0.72	0.68	0.78	0.79
lung_small	0.57	0.61	0.80	0.87	0.82	0.90	**0.93**	0.90	0.90	0.76
lymphoma	0.44	0.50	0.60	0.74	0.69	0.69	**0.76**	0.75	0.69	0.74
GISETTE	0.49	0.51	0.52	**0.54**	0.50	0.51	0.50	0.53	0.50	0.49
Isolet	0.32	0.49	0.72	0.73	0.78	0.78	**0.83**	0.81	0.82	0.83
MADELON	0.61	**0.78**	0.58	0.74	0.64	0.66	0.62	0.64	0.57	0.63
# bests	4	12	1	15	3	13	10	6	4	12

	Decision Tree									
	10		**50**		**100**		**150**		**200**	
	Inf-FS$_U$	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**
PCMAC	0.53	0.50	0.56	0.69	0.58	0.71	0.57	0.68	0.60	**0.73**
RELATHE	0.49	0.50	0.51	**0.51**	0.49	0.42	0.48	0.51	0.48	0.51
COIL20	0.68	0.81	0.83	0.89	0.85	**0.90**	0.89	0.90	0.90	0.90
ORL	0.36	0.39	0.42	0.48	0.49	0.54	0.59	0.61	0.49	**0.62**
warpAR10P	0.37	0.33	0.46	0.59	0.55	0.59	0.41	0.64	0.68	**0.80**
warpPIE10P	0.74	0.74	0.80	0.73	0.77	0.85	0.76	**0.87**	0.76	0.81
Yale	0.17	0.31	0.26	0.34	0.39	0.42	0.50	0.43	0.43	**0.52**
USPS	0.73	0.72	0.84	0.85	0.85	0.86	0.86	0.86	**0.88**	0.87
colon	0.61	0.64	0.82	0.74	0.76	**0.92**	0.89	0.83	0.85	0.83
GLIOMA	0.34	**0.61**	0.36	0.31	0.35	0.44	0.38	0.31	0.31	0.44
lung	0.44	0.70	0.75	0.71	0.87	0.70	**0.90**	0.73	0.71	0.79
lung_small	0.46	0.42	0.58	**0.63**	0.47	0.57	0.52	0.63	0.52	0.49
lymphoma	0.20	**0.69**	0.45	0.55	0.45	0.44	0.63	0.60	0.51	0.62
GISETTE	0.52	0.50	0.44	**0.52**	0.48	0.47	0.50	0.49	0.49	0.48
Isolet	0.27	0.43	0.69	0.67	0.73	0.71	0.74	0.73	**0.78**	0.73
MADELON	0.58	0.66	0.70	**0.81**	0.78	0.79	0.75	0.77	0.74	0.77
# bests	6	10	5	11	5	11	7	9	5	11

out of 16 datasets (i.e. 18.75% of cases). Considering only the scenarios where TFS is the winning feature selection schema, we notice that in 2 cases, the optimal cardinality is equal to 10 and 100, in 4 cases, the optimal cardinality is equal to 50 and 200, in only one case the optimal cardinality is equal to 150. When the Inf-FS$_U$ is used as feature selection algorithm and Decision Tree as classifier, the cross-datasets average Balanced Accuracy score grows from a value equal to 0.47 at cardinality 10, to a value of 0.63 at cardinality 200 with an increase of 16%. When TFS is used as feature selection algorithm and Decision Tree as classifier, the cross-datasets average Balanced Accuracy score grows from a value equals to 0.56 at cardinality 10, to a value of 0.68 at cardinality 200 with an increase of 12%. Finally, we notice that, when Inf-FS$_U$ is used as feature selection algorithm and Decision Tree as classifier, the average maximum drawdown ratio is equal to 22%. Using TFS as feature selection algorithm, instead, the average maximum drawdown ratio is equal to 20%.

Considering the cross-datasets average Balanced Accuracy scores discussed earlier in this Section, we conclude that, independently from the chosen classifier, TFS generally allows to select more informative features, guaranteeing higher learning performances. Both the cross-datasets average Balanced Accuracy score percentage increase and the cross-datasets average maximum drawdown ratio are lower when TFS is chosen as feature selection schema, further certifying an higher stability and an ability to choose higher quality features.

The power of TFS can be further investigated by comparing results obtained by applying the three classification algorithms on the raw datasets (i.e. the datasets containing all the original features) against the best ones obtained by applying the same classifiers combined with the novel feature classification technique presented in this paper. Results are reported in Table 3. When LinearSVM is chosen as a classifier, feature selection turns out to be beneficial on 9 out of 16 datasets (i.e. 56.25% of cases). In 7 cases, TFS is the optimal feature classification approach; in the case of the 'colon' dataset, even if Inf-FS$_U$ is the optimal feature selection approach, results achieved using TFS are still better than the ones obtained on the raw dataset; in the case of 'GLIOMA' dataset, Inf-FS$_U$ is the optimal feature selection approach and results achieved using TFS are lower than the ones obtained on the raw dataset. When KNN is chosen as a classifier, feature selection is beneficial on 9 out of 16 datasets (i.e. 56.25% of cases). In 4 cases, TFS is the optimal feature classification approach; in the case of 'USPS', 'colon', 'lung_small' and 'lymphoma' dataset, even if Inf-FS$_U$ is the optimal feature selection approach, results achieved using TFS are still better than the ones obtained on the raw dataset; in the case of 'COIL20' dataset,

Table 3. Out-of-sample Balanced Accuracy scores obtained by LinearSVM, KNN and Decision Tree classifier on the raw datasets (i.e. the datasets containing all the original features). We boldly highlight the entries where a TFS improves the classifier's performance. We do not highlight the entries where a classifier performs better on the raw dataset (i.e. where feature selection algorithms Inf-FS$_U$ and TFS are not effective). The * symbol highlights the scenarios where the optimal feature selection schema is Inf-FS$_U$ but TFS, in combination with the classifier, still outperforms the classifier on the raw dataset. The † symbol highlights the scenarios where the optimal feature selection schema is Inf-FS$_U$ while TFS, in combination with the classifier, cannot outperform the classifier on the raw dataset.

	LinearSVM	**KNN**	**Decision Tree**
PCMAC	0.83	0.71	0.90
RELATHE	0.84	0.77	0.85
COIL20	0.97	0.96†	**0.87**
ORL	0.97	0.82	0.68
warpAR10P	1.00	0.53	**0.66**
warpPIE10P	**1.00**	**0.84**	**0.84**
Yale	0.82	**0.46**	**0.44**
USPS	**0.93**	0.95*	0.87*
colon	0.80*	0.73*	**0.80**
GLIOMA	0.56†	0.78	**0.44**
lung	**0.93**	**0.76**	0.66*
lung_small	**0.79**	0.76*	**0.63**
lymphoma	**0.93**	0.69*	**0.61**
GISETTE	0.98	0.96	0.92
Isolet	0.93	0.84	0.79
MADELON	**0.58**	**0.51**	**0.74**

Inf-FS$_U$ is the optimal feature selection approach and results achieved using TFS are lower than the ones obtained on the raw dataset. Finally, when Decision Tree is chosen as a classifier, feature selection is beneficial on 11 out of 16 datasets (i.e. 68.75% of cases). In 9 cases, TFS is the optimal feature classification approach; in the case of 'USPS' and 'colon' dataset, even if Inf-FS$_U$ is the optimal feature selection approach, results achieved using TFS are still better than the ones obtained on the raw dataset.

Looking at the results reported above, we notice that the application domains where TFS is more compelling are the ones where the tabular format is the natural data format (i.e. biological data and artificial data). This finding is not unexpected. Application domains such as text, face images and spoken letters data would require a more complex data pre-processing pipeline (e.g. encoding) and specific deep learning-based classification algorithms (e.g. convolutional and recurrent neural networks). A deeper analysis of this aspect is left for the upcoming TFS-centered research.

The statistical significance of results discussed earlier in this Section is assessed in Appendix G. Looking at the entries of Tables 2 where TFS defines a new state-of-the-art, we state that (i) when LinearSVM is used as classifier, TFS is statistically different from Inf-FS$_U$ in 8 out of 14 cases (57%); (ii) when KNN is used as classifier, TFS is statistically different from Inf-FS$_U$ in 3 out of 10 cases (30%); (iii) when Decision Tree is used as classifier, TFS is statistically different from Inf-FS$_U$ in 4 out of 13 cases (31%). There is only one dataset (i.e. 'GISETTE') where TFS is statistically different from Inf-FS$_U$ independently from the classifier: in all the other cases, results are dependent on the choice of the classifier.

4. Conclusions

In this work, we combine the power of state-of-the-art IFNs, and instruments from network science to develop a novel unsupervised, graph-based filter method for feature selection. Features are represented as nodes in a TMFG, and their relevance is assessed by studying their relative position inside the network. Exploiting topological properties of the network used to represent meaningful interactions among features, we propose a physics-informed (i.e. we use instruments from complexity science) feature selection model that is highly flexible, computationally cheap, fully explainable and remarkably intuitive. To prove the effectiveness of the proposed methodology, we test it against the current state-of-the-art counterpart (i.e. Inf-FS$_U$) on 16 benchmark datasets belonging to different applicative domains. Employing a Linear Support Vector classifier, a k-Nearest Neighbors classifier and a Decision Tree classifier, we show how our algorithm achieves top performances on most benchmark datasets, redefining the current state-of-the-art on a significant number of them. The proposed methodology demonstrates effectiveness in conditions where the amount of training data largely varies. Compared to its main alternative, TFS has a lower computational complexity and provides a much more intuitive overview of the feature selection process. Thanks to the possibility of studying the relative position of nodes in the network in many different ways (i.e. choosing different centrality measures or defining new ones), TFS is highly versatile and fully adaptable to input data. It is worth noting that the current work is a foundational one. It presents three aspects that are unsatisfactory and we plan to cover in the future: (i) the need to explicitly specify the cardinality of the subset of relevant features is limiting and requires an a priori knowledge of the applicative domain or, at least, an extended search for the optimal realization of this hyper-parameter; (ii) the usage of classic correlation measures in the TMFG's building process prevents from the possibility to handle problems with mixed type of features (continuous-categorical, categorical-categorical); (iii) TMFG is non-differentiable and this prevents from a direct integration with advanced Deep Learning-based architectures. More generally, this study points to many future directions spanning from the development of data-centred measures to assess features relevance, to the possibility of replicating the potential of this method and inferring it through automated learning techniques. The first steps have been taken in the latter research direction by introducing a new and potentially groundbreak-

ing family of Neural Networks called Homological Neural Networks (Wang et al., 2023).

Acknowledgements

Both the authors acknowledge Agne Scalchi, Silvia Bartolucci and Paolo Barucca for for the fruitful discussions on foundational topics related to this work. Both the authors acknowledge the ICML TAG-ML 2023 workshop's organising committee and the reviewers for the useful comments that improved the quality of the paper. The author, T.A., acknowledges the financial support from ESRC (ES/K002309/1), EPSRC (EP/P031730/1) and EC (H2020-ICT-2018-2 825215).

References

Feature selection datasets, arizona state university. https://jundongl.github.io/scikit-feature/datasets.html, 2020. Accessed: 2022-09-19.

Infinite feature selection. https://github.com/fullyz/Infinite-Feature-Selection/tree/master/data, 2020. Accessed: 2022-09-19.

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., and Levine, A. J. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 96(12): 6745–6750, 1999.

Aste, T. Topological regularization with information filtering networks. *Information Sciences*, 608:655–669, 2022.

Aste, T., Di Matteo, T., and Hyde, S. Complex networks on hyperbolic surfaces. *Physica A: Statistical Mechanics and its Applications*, 346(1-2):20–26, 2005.

Baldi, P., Brunak, S., Chauvin, Y., Andersen, C. A., and Nielsen, H. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16 (5):412–424, 2000.

Barfuss, W., Massara, G. P., Di Matteo, T., and Aste, T. Parsimonious modeling with information filtering networks. *Physical Review E*, 94(6):062306, 2016.

Belhumeur, P. N., Hespanha, J. P., and Kriegman, D. J. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):711–720, 1997.

Briola, A. and Aste, T. Dependency structures in cryptocurrency market from high to low frequency. *arXiv preprint arXiv:2206.03386*, 2022.

Briola, A., Vidal-Tomás, D., Wang, Y., and Aste, T. Anatomy of a stablecoin's failure: The terra-luna case. *Finance Research Letters*, pp. 103358, 2022.

Chandrashekar, G. and Sahin, F. A survey on feature selection methods. *Computers & Electrical Engineering*, 40 (1):16–28, 2014.

Christensen, A. P. Networktoolbox: Methods and measures for brain, cognitive, and psychometric network analysis in r. *R J.*, 10(2):422, 2018.

Christensen, A. P., Kenett, Y. N., Aste, T., Silvia, P. J., and Kwapil, T. R. Network structure of the wisconsin schizotypy scales–short forms: Examining psychometric network filtering approaches. *Behavior Research Methods*, 50(6):2531–2550, 2018.

Danoff, J. S., Wroblewski, K. L., Graves, A. J., Quinn, G. C., Perkeybile, A. M., Kenkel, W. M., Lillard, T. S., Parikh, H. I., Golino, H. F., Gregory, S. G., et al. Genetic, epigenetic, and environmental factors controlling oxytocin receptor gene expression. *Clinical epigenetics*, 13(1): 1–16, 2021.

Dietterich, T. G. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7):1895–1923, 1998.

Donoho, D. L. et al. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS math challenges lecture*, 1(2000):32, 2000.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Eichelberger, H., Qin, C., and Schmid, K. Experiences with the model-based generation of big data pipelines. *Datenbanksysteme für Business, Technologie und Web (BTW 2017)-Workshopband*, 2017.

Ge, R., Zhou, M., Luo, Y., Meng, Q., Mai, G., Ma, D., Wang, G., and Zhou, F. Mctwo: a two-step feature selection algorithm based on maximal information coefficient. *BMC bioinformatics*, 17(1):1–14, 2016.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, 286(5439):531–537, 1999.

Gorodkin, J. Comparing two k-category assignments by a k-category correlation coefficient. *Computational biology and chemistry*, 28(5-6):367–374, 2004.

Guyon, I., Li, J., Mader, T., Pletscher, P. A., Schneider, G., and Uhr, M. Competitive baseline methods set new

standards for the nips 2003 feature selection benchmark. *Pattern recognition letters*, 28(12):1438–1444, 2007.

Guyon, I., Gunn, S., Nikravesh, M., and Zadeh, L. A. *Feature extraction: foundations and applications*, volume 207. Springer, 2008.

Han, J., Pei, J., and Tong, H. *Data mining: concepts and techniques*. Morgan kaufmann, 2022.

Huang, S. H. Supervised feature selection: A tutorial. *Artif. Intell. Res.*, 4(2):22–37, 2015.

Hull, J. J. A database for handwritten text recognition research. *IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

Hutter, J., Slator, P. J., Jackson, L., Gomes, A. D. S., Ho, A., Story, L., O'Muircheartaigh, J., Teixeira, R. P., Chappell, L. C., Alexander, D. C., et al. Multi-modal functional mri to explore placental function over gestation. *Magnetic resonance in medicine*, 81(2):1191–1204, 2019.

Jurman, G., Riccadonna, S., and Furlanello, C. A comparison of mcc and cen error measures in multi-class prediction. 2012.

Kelleher, J. D., Mac Namee, B., and D'arcy, A. *Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies*. MIT press, 2020.

Koller, D. and Friedman, N. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Lang, K. Newsweeder: Learning to filter netnews. In *Machine Learning Proceedings 1995*, pp. 331–339. Elsevier, 1995.

Lee, C.-Y. Correlations among centrality measures in complex networks. *arXiv preprint physics/0605220*, 2006.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6):94, 2018.

Mantegna, R. N. Hierarchical structure in financial markets. *The European Physical Journal B-Condensed Matter and Complex Systems*, 11(1):193–197, 1999.

Massara, G. P., Di Matteo, T., and Aste, T. Network filtering for big data: Triangulated maximally filtered graph. *Journal of complex Networks*, 5(2):161–178, 2017.

Matthews, B. W. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.

Miao, J. and Niu, L. A survey on feature selection. *Procedia Computer Science*, 91:919–926, 2016.

Miner, G., Elder IV, J., Fast, A., Hill, T., Nisbet, R., and Delen, D. *Practical text mining and statistical analysis for non-structured text data applications*. Academic Press, 2012.

Mingqiang, Y., Kidiyo, K., Joseph, R., et al. A survey of shape feature extraction techniques. *Pattern recognition*, 15(7):43–90, 2008.

Mosley, L. A balanced approach to the multi-class imbalance problem. *Doctor of Philosophy Thesis, Iowa State University of Science and Technology, USA*, 2013.

Nene, S. A., Nayar, S. K., Murase, H., et al. Columbia object image library (coil-100). 1996.

Papadimitriou, C. H. and Steiglitz, K. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

Patterson, D., Gonzalez, J., Le, Q., Liang, C., Munguia, L.-M., Rothchild, D., So, D., Texier, M., and Dean, J. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*, 2021.

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

Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., and Liao, Q. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: a review. *International Journal of Automation and Computing*, 14(5):503–519, 2017.

Procacci, P. F. and Aste, T. Portfolio optimization with sparse multivariate modeling. *Journal of Asset Management*, 23(6):445–465, 2022.

Roffo, G., Melzi, S., and Cristani, M. Infinite feature selection. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4202–4210, 2015.

Roffo, G., Melzi, S., Castellani, U., Vinciarelli, A., and Cristani, M. Infinite feature selection: a graph-based feature filtering approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(12):4396–4410, 2020.

Samaria, F. S. and Harter, A. C. Parameterisation of a stochastic model for human face identification. In *Proceedings of 1994 IEEE workshop on applications of computer vision*, pp. 138–142. IEEE, 1994.

Seabrook, I., Caccioli, F., and Aste, T. Quantifying impact and response in markets using information filtering networks. *Journal of Physics: Complexity*, 3(2):025004, 2022.

Sheikhpour, R., Sarram, M. A., Gharaghani, S., and Chahooki, M. A. Z. A survey on semi-supervised feature selection methods. *Pattern Recognition*, 64:141–158, 2017.

Shirokikh, O., Pastukhov, G., Boginski, V., and Butenko, S. Computational study of the us stock market evolution: a rank correlation-based network model. *Computational Management Science*, 10(2):81–103, 2013.

Sim, T., Baker, S., and Bsat, M. The cmu pose, illumination, and expression (pie) database. In *Proceedings of fifth IEEE international conference on automatic face gesture recognition*, pp. 53–58. IEEE, 2002.

Solorio-Fernández, S., Carrasco-Ochoa, J. A., and Martínez-Trinidad, J. F. A review of unsupervised feature selection methods. *Artificial Intelligence Review*, 53(2):907–948, 2020.

Song, W.-M., Aste, T., and Di Matteo, T. Correlation-based biological networks. In *Complex Systems II*, volume 6802, pp. 226–236. SPIE, 2008.

Song, W.-M., Di Matteo, T., and Aste, T. Nested hierarchies in planar graphs. *Discrete Applied Mathematics*, 159(17): 2135–2146, 2011.

Song, W.-M., Di Matteo, T., and Aste, T. Hierarchical information clustering by means of topologically embedded graphs. *PloS one*, 7(3):e31929, 2012.

Strubell, E., Ganesh, A., and McCallum, A. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*, 2019.

Tumminello, M., Aste, T., Di Matteo, T., and Mantegna, R. N. A tool for filtering information in complex systems. *Proceedings of the National Academy of Sciences*, 102 (30):10421–10426, 2005.

Valente, T. W., Coronges, K., Lakon, C., and Costenbader, E. How correlated are network centrality measures? *Connections (Toronto, Ont.)*, 28(1):16, 2008.

Vidal-Tomás, D., Briola, A., and Aste, T. Ftx's downfall and binance's consolidation: the fragility of centralized digital finance. *arXiv preprint arXiv:2302.11371*, 2023.

Wang, Y. and Aste, T. Dynamic portfolio optimization with inverse covariance clustering. *Expert Systems with Applications*, pp. 118739, 2022a.

Wang, Y. and Aste, T. Sparsification and filtering for spatial-temporal gnn in multivariate time-series. *arXiv preprint arXiv:2203.03991*, 2022b.

Wang, Y., Briola, A., and Aste, T. Homological neural networks: A sparse architecture for multivariate complexity, 2023.

Wu, D., Zhu, L., Xu, X., Sakr, S., Sun, D., and Lu, Q. Building pipelines for heterogeneous execution environments for big data processing. *IEEE Software*, 33(2):60–67, 2016.

Zeng, X. and Martinez, T. R. Distribution-balanced stratified cross-validation for accuracy estimation. *Journal of Experimental & Theoretical Artificial Intelligence*, 12(1): 1–12, 2000.

A. Benchmark Datasets

To prove TFS' effectiveness, we extensively test it on 16 benchmark datasets (all in a tabular format) belonging to different application domains. For each dataset, Table 4 reports, respectively, the name, the application domain, the reference paper (or website), the downloading source, the number of features, the number of samples, the number of classes and the split dynamics (see Appendix B for further details on this last point).

Table 4. Benchmark datasets used to compare feature selection algorithms considered in the current work. The order of appearance is inherited from (ASU, 2020).

Name	Category	Reference	Source	# Features	# Samples	Classes	Split Provided
PCMAC	Text Data	(Lang, 1995)	(ASU, 2020)	3289	1943	binary	false
RELATHE	Text Data	(Lang, 1995)	(ASU, 2020)	4322	1427	binary	false
COIL20	Face Images Data	(Nene et al., 1996)	(ASU, 2020)	1024	1440	multi-class	false
ORL	Face Images Data	(Samaria & Harter, 1994)	(ASU, 2020)	1024	400	multi-class	false
warpAR10P	Face Images Data	(Li et al., 2018)	(ASU, 2020)	2400	130	multi-class	false
warpPIE10P	Face Images Data	(Sim et al., 2002)	(ASU, 2020)	2420	210	multi-class	false
Yale	Face Images Data	(Belhumeur et al., 1997)	(ASU, 2020)	1024	165	multi-class	false
USPS	Hand Written Images Data	(Hull, 1994)	(Git, 2020)	256	9298	multi-class	false
colon	Biological Data	(Alon et al., 1999)	(Git, 2020)	2000	62	binary	false
GLIOMA	Biological Data	(Li et al., 2018)	(ASU, 2020)	4434	50	multi-class	false
lung	Biological Data	(Git, 2020)	(Git, 2020)	3312	203	multi-class	false
lung_small	Biological Data	(Git, 2020)	(Git, 2020)	325	73	multi-class	false
lymphoma	Biological Data	(Golub et al., 1999)	(ASU, 2020)	4026	96	multi-class	false
GISETTE	Digits Data	(Guyon et al., 2007)	(Dua & Graff, 2017)	5000	7000	binary	true
Isolet	Spoken Letters Data	(Li et al., 2018)	(ASU, 2020)	617	1560	multi-class	false
MADELON	Artificial Data	(Guyon et al., 2007)	(Dua & Graff, 2017)	500	2600	binary	true

We distinguish among 7 different application domains (i.e. text data, face images data, hand written images data, biological data, digits data, spoken letters data and artificial data). Categories follow the taxonomy in (Li et al., 2018). The average number of features is 2248. The dataset with the lowest number of features is 'USPS' (i.e. 256). The dataset with the largest number of features is 'GISETTE' (i.e. 5000). The average number of samples is 1666. The dataset with the lowest number of samples is 'GLIOMA' (i.e. 50), while the one with the largest number of samples is 'USPS' (i.e. 9298). 5 of the considered datasets are binary, while 11 are multi-class.

B. Data Pipeline

We design the data pre-processing pipeline as consisting of 3 different steps: (i) data reading and format unification; (ii) training/validation/test splitting; (iii) constant features pruning. The first step allows us to read data and unify formats coming from different sources. The second step consists of the training/validation/test splitting. Depending on the source, training/test split could be provided or not. The two datasets 'GISETTE' and 'MADELON' come with a provided training/validation/test split. In both cases, the data source (i.e. (Dua & Graff, 2017)) does not provide test labels. Because of this, we use the validation set for testing. For all the other datasets, 70% of the raw dataset is used as a training and validation set, while 30% is used as a test set. We use a stratified splitting procedure to ensure that each set contains, for each target class, approximately the same percentage of samples as per in the raw dataset. In the third step, non-informative, constant covariates are detected on the training set and permanently removed from the training, validation and test set.

Table 5 reports datasets' specifics after the preprocessing step. Looking at it, we remark that most considered datasets are not affected by the constant features filtering step. The only 4 datasets which are reduced in the number of covariates are 'PCMAC' (with a reduction of 0.07%), 'RELATHE' (with a reduction of 0.03%), 'GLIOMA' (with a reduction of 0.03%) and 'GISETTE' (with a reduction of 0.9%). Table 5 also reports the training and test dataset's labels' distributions. Datasets are generally balanced; the main exceptions are:

- 'USPS': classes 1 and 2 are over-represented compared to the other classes.
- 'colon': class 1 is under-represented compared to the other class.
- 'GLIOMA': class 2 is under-represented compared to other classes.
- 'lung': class 1 is over-represented compared to the other classes.
- 'lung_small': classes 1, 2, 3 and 5 are under-represented compared to the other classes.
- 'lymphoma': class 1 is over-represented compared to the other classes.

Table 5. Datasets' specifics after the preprocessing step. For each benchmark dataset, we report the number of features, the number of samples and the labels' distribution for training and test data. The labels' distribution entry consists of a tuple for each class. The first element of the tuple represents the class itself, while the second represents the number of samples with that label.

Dataset	Training			Test		
	# Features	# Samples	Labels' Distribution	# Features	# Samples	Labels' Distribution
PCMAC	3287	1360	(1, 687), (2, 673)	3287	583	(1, 295), (2, 288)
RELATHE	4321	998	(1, 545), (2, 453)	4321	429	(1, 234), (2, 195)
COIL20	1024	1008	(1, 50), (2, 51), (3, 50), (4, 50), (5, 50), (6, 50), (7, 51), (8, 50), (9, 51), (10, 50), (11, 51), (12, 50), (13, 50), (14, 51), (15, 50), (16, 50), (17, 50), (18, 51), (19, 51), (20, 51)	1024	432	(1, 22), (2, 21), (3, 22), (4, 22), (5, 22), (6, 22), (7, 21), (8, 22), (9, 21), (10, 22), (11, 21), (12, 22), (13, 22), (14, 21), (15, 22), (16, 22), (17, 22), (18, 21), (19, 21), (20, 21)
ORL	1024	280	(1, 7), (2, 7), (3, 7), (4, 7), (5, 7), (6, 7), (7, 7), (8, 7), (9, 7), (10, 7), (11, 7), (12, 7), (13, 7), (14, 7), (15, 7), (16, 7), (17, 7), (18, 7), (19, 7), (20, 7), (21, 7), (22, 7), (23, 7), (24, 7), (25, 7), (26, 7), (27, 7), (28, 7), (29, 7), (30, 7), (31, 7), (32, 7), (33, 7), (34, 7), (35, 7), (36, 7), (37, 7), (38, 7), (39, 7), (40, 7)	1024	120	(1, 3), (2, 3), (3, 3), (4, 3), (5, 3), (6, 3), (7, 3), (8, 3), (9, 3), (10, 3), (11, 3), (12, 3), (13, 3), (14, 3), (15, 3), (16, 3), (17, 3), (18, 3), (19, 3), (20, 3), (21, 3), (22, 3), (23, 3), (24, 3), (25, 3), (26, 3), (27, 3), (28, 3), (29, 3), (30, 3), (31, 3), (32, 3), (33, 3), (34, 3), (35, 3), (36, 3), (37, 3), (38, 3), (39, 3), (40, 3)
warpAR10P	2400	91	(1, 9), (2, 9), (3, 10), (4, 9), (5, 9), (6, 9), (7, 9), (8, 9), (9, 9), (10, 9)	2400	39	(1, 4), (2, 4), (3, 3), (4, 4), (5, 4), (6, 4), (7, 4), (8, 4), (9, 4), (10, 4)
warpPIE10P	2420	147	(1, 14), (2, 15), (3, 15), (4, 14), (5, 15), (6, 14), (7, 15), (8, 15), (9, 15), (10, 15)	2420	63	(1, 7), (2, 6), (3, 6), (4, 7), (5, 6), (6, 7), (7, 6), (8, 6), (9, 6), (10, 6)
Yale	1024	115	(1, 7), (2, 8), (3, 8), (4, 7), (5, 8), (6, 7), (7, 8), (8, 8), (9, 8), (10, 8), (11, 8), (12, 7), (13, 7), (14, 8), (15, 8)	1024	50	(1, 4), (2, 3), (3, 3), (4, 4), (5, 3), (6, 4), (7, 3), (8, 3), (9, 3), (10, 3), (11, 3), (12, 4), (13, 4), (14, 3), (15, 3)
USPS	256	6508	(1, 1087), (2, 888), (3, 650), (4, 577), (5, 596), (6, 501), (7, 584), (8, 554), (9, 496), (10, 575)	256	2790	(1, 466), (2, 381), (3, 279), (4, 247), (5, 256), (6, 215), (7, 250), (8, 238), (9, 212), (10, 246)
colon	2000	43	(-1, 28), (1, 15)	2000	19	(-1, 12), (1, 7)
GLIOMA	4433	35	(1, 10), (2, 5), (3, 10), (4, 10)	4433	15	(1, 4), (2, 2), (3, 4), (4, 5)
lung	3312	142	(1, 97), (2, 12), (3, 15), (4, 14), (5, 4)	3312	61	(1, 42), (2, 5), (3, 6), (4, 6), (5, 2)
lung_small	325	51	(1, 4), (2, 3), (3, 4), (4, 11), (5, 5), (6, 9), (7, 15)	325	22	(1, 2), (2, 2), (3, 1), (4, 5), (5, 2), (6, 4), (7, 6)
lymphoma	4026	67	(1, 32), (2, 7), (3, 6), (4, 8), (5, 4), (6, 4), (7, 3), (8, 1), (9, 2)	4026	29	(1, 14), (2, 3), (3, 3), (4, 3), (5, 2), (6, 2), (7, 1), (8, 1)
GISETTE	4955	6000	(-1.0, 3000), (1.0, 3000)	4955	1000	(-1.0, 500), (1.0, 500)
Isolet	617	1092	(1, 42), (2, 42), (3, 42), (4, 42), (5, 42), (6, 42), (7, 42), (8, 42), (9, 42), (10, 42), (11, 42), (12, 42), (13, 42), (14, 42), (15, 42), (16, 42), (17, 42), (18, 42), (19, 42), (20, 42), (21, 42), (22, 42), (23, 42), (24, 42), (25, 42), (26, 42)	617	468	(1, 18), (2, 18), (3, 18), (4, 18), (5, 18), (6, 18), (7, 18), (8, 18), (9, 18), (10, 18), (11, 18), (12, 18), (13, 18), (14, 18), (15, 18), (16, 18), (17, 18), (18, 18), (19, 18), (20, 18), (21, 18), (22, 18), (23, 18), (24, 18), (25, 18), (26, 18)
MADELON	500	2000	(-1.0, 1000), (1.0, 1000)	500	600	(-1.0, 300), (1.0, 300)

C. Triangulated Maximally Filtered Graph

The building process of the Triangulated Maximally Filtered Graph (TMFG) (Massara et al., 2017) is based on a simple topological move that preserves the property of planarity: it adds one node to the centre of three-nodes cliques by using a score function that maximises the sum of the weights of the three edges connecting the existing vertices. This addition transforms three-nodes cliques (i.e. triangles) into four-nodes cliques (i.e. tetrahedrons) characterised by a chord that is not part of the clique but connects two nodes in the clique, forming two triangles and generating a chordal network (Christensen et al., 2018). The resulting network has $3n - 6$ edges and is composed of three- and four-nodes cliques. TMFG has two relevant advantages: (i) it can be used to generate sparse probabilistic models as a form of topological regularization (Aste, 2022) and (ii) it is computationally efficient. On the other hand, the two main limitations of chordal networks are that (i) they may add unnecessary edges to satisfy the property of chordality and (ii) their building cost can vary based on the chosen optimization function.

Algorithm 1 TMFG built on the similarity matrix $\hat{\mathbf{C}}$ to maximise the system's information flow.

input Similarity matrix $\hat{\mathbf{C}} \in \mathbb{R}^{n,n}$ from a set of observations $\{x_{1,1}, \ldots, x_{s,1}\}, \{x_{1,2}, \ldots, x_{s,2}\} \ldots \{x_{1,n}, \ldots, x_{s,n}\}$.
output Sparse adjacency matrix $\boldsymbol{A}$ describing the TMFG.

function MaximumGain ($\hat{\mathbf{C}}$, $\mathcal{V}$, t)
  Initialize a vector of zeros $g \in \mathbb{R}^{1\times n}$;
  for $j \in t$ **do**
    for $v \notin \mathcal{V}$ **do**
      $\hat{\mathbf{C}}_{v,j} = 0$;
    end for
    $g = g \oplus \hat{\mathbf{C}}_{v,j}$;
  end for
  return $\max\{g\}$.
end function

Initialize four empty sets: $\mathcal{C}$ (cliques), $\mathcal{T}$ (triangles), $\mathcal{S}$ (separators) and $\mathcal{V}$ (vertices);
Initialize an adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{n,n}$ with all zeros;
$\mathcal{C}_1 \leftarrow$ tetrahedron, $\{v_1, v_2, v_3, v_4\}$, obtained choosing the 4 entries of $\hat{\mathbf{C}}$ maximising the similarity among features;
$\mathcal{T} \leftarrow$ the four triangular faces in $\mathcal{C}_1 : \{v_1, v_2, v_3\}, \{v_1, v_2, v_4\}, \{v_1, v_3, v_4\}, \{v_2, v_3, v_4\}$;
$\mathcal{V} \leftarrow$ Assign to $\mathcal{V}$ the remaining $n - 4$ vertices not in $\mathcal{C}_1$;
while $\mathcal{V}$ is not empty **do**
  Find the combination of $\{v_a, v_b, v_c\} \in \mathcal{T}$ (i.e. t) and $v_d \in \mathcal{V}$ which maximises MaximumGain($\hat{\mathbf{C}}$, $\mathcal{V}$, t);
  /* $\{v_a, v_b, v_c, v_d\}$ is a new 4-clique $\mathcal{C}$, $\{v_a, v_b, v_c\}$ becomes a separator $\mathcal{S}$, three new triangular faces, $\{v_a, v_b, v_d\}$, $\{v_a, v_c, v_d\}$ and $\{v_b, v_c, v_d\}$ are created */.
  Remove v_d from $\mathcal{V}$;
  Remove $\{v_a, v_b, v_c\}$ from $\mathcal{T}$;
  Add $\{v_a, v_b, v_d\}$, $\{v_a, v_c, v_d\}$ and $\{v_b, v_c, v_d\}$ to $\mathcal{T}$;
end while
For each pair of nodes i, j in $\mathcal{C}$, set $\boldsymbol{A}_{i,j} = 1$;
return $\boldsymbol{A}$.

D. Classification algorithms

As reported in Section 2.5, the meaningfulness of the features' subsets chosen by Inf-FS$_U$ and TFS is evaluated based on the performance achieved by three classification algorithms: (i) Linear Support Vector Classifier; (ii) *k*-Nearest Neighbors Classifier; (iii) Decision Tree Classifier. For all of them, we use the implementation provided by the 'scikit-learn' Python package (Pedregosa et al., 2011). The interested reader is referred to the following links for the implementations:

- Linear Support Vector Classifier: https://github.com/scikit-learn/scikit-learn/blob/98cf537f5/sklearn/svm/_classes.py#L14
- *k*-Nearest Neighbors Classifier: https://github.com/scikit-learn/scikit-learn/blob/98cf537f5/sklearn/neighbors/_classification.py#L24
- Decision Tree Classifier: https://github.com/scikit-learn/scikit-learn/blob/98cf537f5/sklearn/tree/_classes.py#L595

For the current research work we do not significantly change the models' default hyper-parameters. The first adjustment is performed on the Linear Support Vector Classifier's `max_iter` parameter, which is set to 50000. For Linear Support Vector Classifier and Decision Tree Classifier, the `random_seed` parameter is set to 0.

E. Optimal hyper-parameters configurations

For each benchmark dataset, subset's cardinality and classification algorithm described in Section 2.5, Tables 6 and 7, report optimal hyper-parameters configurations.

It is worth mentioning that after the feature selection step, input features are standardised by removing the mean and scaling to unit variance. The standard score of a sample x is hence calculated as:

$$z = \frac{x - \hat{\mu}}{\hat{\sigma}} \tag{7}$$

where $\hat{\mu}$ and $\hat{\sigma}$ are the mean and the standard deviation of the training samples. This step is performed using the `StandardScaler` implementation provided by the 'scikit-learn' Python package (Pedregosa et al., 2011) at the following link: https://github.com/scikit-learn/scikit-learn/blob/98cf537f5/sklearn/preprocessing/_data.py#L644.

During the stratified k-fold cross-validation stage, classes samples are shuffled before splitting into batches. The `random_state` parameter is set to 0.

Hyper-parameters search is performed using a modified, parallel grid search approach. Also in this case, the basic implementation is provided by the 'scikit-learn' Python package (Pedregosa et al., 2011) at the following link: https://github.com/scikit-learn/scikit-learn/blob/98cf537f5/sklearn/model_selection/_search.py#L1031.

Table 6. Subset' size-dependent in-sample optimal hyper-parameters configurations and corresponding balanced accuracy scores for Inf-FS$_U$.

		10			**50**			**100**			**150**			**200**		
		α	θ	**score**	α	θ	**score**	α	θ	**score**	α	θ	**score**	α	θ	**score**
PCMAC	LinearSVM	0.90	0.80	0.55	0.80	0.10	0.57	0.50	0.10	0.60	0.40	0.10	0.61	0.90	0.10	0.64
	KNN	0.20	0.10	0.52	0.40	0.90	0.58	0.40	0.10	0.59	0.50	0.10	0.61	0.50	0.50	0.61
	Decision Tree	0.30	0.10	0.55	0.90	0.10	0.60	0.80	0.30	0.62	0.90	0.10	0.64	0.40	0.70	0.65
RELATHE	LinearSVM	0.40	0.10	0.59	0.30	0.10	0.69	0.40	0.10	0.75	0.70	0.10	0.75	0.30	0.10	0.75
	KNN	0.80	0.10	0.59	0.30	0.30	0.67	0.50	0.80	0.72	0.40	0.10	0.72	0.40	0.10	0.73
	Decision Tree	0.40	0.10	0.58	0.40	0.90	0.67	0.50	0.30	0.72	0.60	0.30	0.74	0.40	0.20	0.74
COIL20	LinearSVM	0.70	0.90	0.50	0.60	0.50	0.74	0.60	0.10	0.84	0.50	0.90	0.89	0.10	0.30	0.91
	KNN	0.70	0.10	0.67	0.70	0.10	0.85	0.50	0.10	0.89	0.50	0.10	0.92	0.10	0.10	0.94
	Decision Tree	0.70	0.10	0.66	0.70	0.30	0.82	0.40	0.80	0.83	0.20	0.80	0.85	0.10	0.30	0.86
ORL	LinearSVM	0.70	0.40	0.35	0.90	0.10	0.63	0.90	0.50	0.74	0.90	0.90	0.77	0.90	0.10	0.78
	KNN	0.70	0.40	0.29	0.90	0.10	0.50	0.90	0.50	0.55	0.90	0.30	0.62	0.10	0.10	0.65
	Decision Tree	0.70	0.10	0.30	0.90	0.30	0.40	0.90	0.80	0.43	0.40	0.20	0.43	0.70	0.20	0.45
warpAR10P	LinearSVM	0.70	0.10	0.39	0.10	0.40	0.61	0.40	0.50	0.71	0.60	0.60	0.78	0.20	0.30	0.82
	KNN	0.70	0.10	0.36	0.90	0.30	0.37	0.60	0.90	0.38	0.10	0.90	0.38	0.10	0.80	0.37
	Decision Tree	0.60	0.10	0.39	0.70	0.60	0.46	0.40	0.80	0.47	0.50	0.10	0.50	0.10	0.50	0.57
warpPIE10P	LinearSVM	0.10	0.30	0.78	0.70	0.10	0.97	0.80	0.10	0.99	0.80	0.10	0.99	0.10	0.10	0.99
	KNN	0.10	0.10	0.64	0.10	0.10	0.81	0.10	0.10	0.81	0.70	0.30	0.82	0.40	0.10	0.85
	Decision Tree	0.70	0.20	0.66	0.10	0.70	0.76	0.10	0.20	0.79	0.50	0.40	0.83	0.40	0.50	0.83
Yale	LinearSVM	0.30	0.10	0.20	0.10	0.70	0.33	0.10	0.10	0.49	0.80	0.30	0.55	0.90	0.10	0.68
	KNN	0.70	0.10	0.25	0.50	0.10	0.34	0.10	0.10	0.36	0.30	0.10	0.42	0.40	0.20	0.49
	Decision Tree	0.90	0.10	0.31	0.60	0.40	0.39	0.30	0.40	0.40	0.90	0.20	0.46	0.80	0.80	0.48
USPS	LinearSVM	0.90	0.10	0.71	0.90	0.10	0.89	0.90	0.10	0.91	0.70	0.40	0.92	0.10	0.10	0.92
	KNN	0.90	0.10	0.76	0.90	0.10	0.93	0.90	0.10	0.96	0.80	0.10	0.96	0.10	0.10	0.95
	Decision Tree	0.90	0.10	0.70	0.90	0.10	0.83	0.90	0.10	0.85	0.40	0.10	0.86	0.20	0.80	0.86
colon	LinearSVM	0.60	0.10	0.72	0.40	0.50	0.81	0.60	0.90	0.78	0.80	0.50	0.76	0.90	0.40	0.81
	KNN	0.70	0.10	0.74	0.80	0.40	0.80	0.80	0.10	0.80	0.80	0.10	0.86	0.70	0.10	0.83
	Decision Tree	0.60	0.10	0.72	0.80	0.90	0.81	0.80	0.40	0.83	0.30	0.70	0.79	0.20	0.90	0.83
GLIOMA	LinearSVM	0.90	0.10	0.55	0.10	0.20	0.62	0.60	0.10	0.63	0.60	0.20	0.70	0.70	0.10	0.67
	KNN	0.90	0.10	0.50	0.90	0.10	0.68	0.90	0.10	0.70	0.90	0.60	0.69	0.90	0.10	0.69
	Decision Tree	0.80	0.50	0.53	0.20	0.10	0.68	0.50	0.10	0.59	0.30	0.80	0.58	0.40	0.40	0.58
lung	LinearSVM	0.90	0.10	0.45	0.90	0.10	0.78	0.80	0.10	0.90	0.80	0.10	0.93	0.60	0.10	0.88
	KNN	0.90	0.10	0.36	0.80	0.10	0.70	0.90	0.10	0.75	0.80	0.10	0.81	0.70	0.10	0.75
	Decision Tree	0.90	0.10	0.32	0.80	0.10	0.56	0.60	0.30	0.69	0.60	0.80	0.71	0.90	0.10	0.76
lung_small	LinearSVM	0.70	0.10	0.62	0.10	0.10	0.72	0.10	0.90	0.71	0.90	0.80	0.76	0.10	0.40	0.75
	KNN	0.70	0.10	0.67	0.30	0.50	0.80	0.30	0.10	0.83	0.40	0.50	0.83	0.10	0.10	0.74
	Decision Tree	0.10	0.80	0.55	0.10	0.40	0.60	0.30	0.90	0.61	0.30	0.30	0.64	0.20	0.10	0.69
lymphoma	LinearSVM	0.90	0.10	0.38	0.60	0.10	0.52	0.70	0.10	0.65	0.60	0.10	0.66	0.30	0.90	0.71
	KNN	0.90	0.90	0.27	0.90	0.10	0.44	0.80	0.50	0.61	0.70	0.10	0.62	0.80	0.10	0.66
	Decision Tree	0.20	0.60	0.33	0.80	0.70	0.51	0.70	0.70	0.50	0.60	0.20	0.55	0.60	0.50	0.60
GISETTE	LinearSVM	0.90	0.10	0.85	0.90	0.10	0.91	0.80	0.10	0.92	0.80	0.10	0.93	0.70	0.10	0.93
	KNN	0.90	0.10	0.86	0.90	0.10	0.94	0.90	0.10	0.94	0.80	0.10	0.94	0.80	0.10	0.94
	Decision Tree	0.90	0.10	0.80	0.80	0.50	0.88	0.80	0.50	0.89	0.90	0.60	0.90	0.80	0.10	0.90
Isolet	LinearSVM	0.60	0.10	0.31	0.80	0.10	0.72	0.90	0.10	0.79	0.90	0.30	0.86	0.90	0.10	0.86
	KNN	0.60	0.10	0.27	0.90	0.10	0.70	0.90	0.50	0.77	0.90	0.10	0.80	0.80	0.30	0.81
	Decision Tree	0.80	0.10	0.24	0.90	0.40	0.68	0.80	0.50	0.71	0.90	0.30	0.75	0.90	0.70	0.75
MADELON	LinearSVM	0.60	0.10	0.61	0.80	0.10	0.60	0.70	0.10	0.59	0.30	0.10	0.57	0.30	0.10	0.55
	KNN	0.80	0.10	0.71	0.80	0.10	0.61	0.90	0.10	0.59	0.90	0.10	0.57	0.80	0.10	0.57
	Decision Tree	0.80	0.10	0.70	0.80	0.10	0.70	0.80	0.80	0.75	0.70	0.70	0.75	0.60	0.60	0.74

Table 7. Subset' size-dependent in-sample optimal hyper-parameters configurations and corresponding balanced accuracy scores for TFS.

		10				50				100				150				200			
		metric	**square**	α	**score**	**metric**	**square**	α	**score**	**metric**	**square**	α	**score**	**metric**	**square**	α	**score**	**metric**	**square**	α	**score**
PCMAC	LinearSVM	Spearman	Square	/	0.56	Pearson	Normal	/	0.62	Pearson	Normal	/	0.64	Pearson	Square	/	0.66	Energy	Normal	0.30	0.69
	KNN	Pearson	Normal	/	0.52	Pearson	Normal	/	0.60	Pearson	Normal	/	0.59	Pearson	Square	/	0.62	Energy	Normal	0.40	0.64
	Decision Tree	Spearman	Square	/	0.56	Pearson	Normal	/	0.63	Pearson	Square	/	0.64	Pearson	Square	/	0.66	Pearson	Normal	/	0.68
RELATHE	LinearSVM	Spearman	Square	/	0.53	Energy	Normal	0.70	0.62	Energy	Normal	0.70	0.69	Energy	Normal	0.90	0.70	Energy	Normal	0.90	0.71
	KNN	Energy	Normal	0.30	0.53	Pearson	Normal	/	0.61	Energy	Normal	0.80	0.64	Energy	Normal	0.30	0.66	Energy	Normal	0.60	0.68
	Decision Tree	Energy	Normal	0.30	0.53	Energy	Normal	0.60	0.60	Spearman	Normal	/	0.64	Energy	Normal	0.80	0.68	Spearman	Square	/	0.67
COIL20	LinearSVM	Pearson	Square	/	0.71	Pearson	Square	/	0.91	Energy	Normal	0.20	0.93	Pearson	Square	/	0.95	Pearson	Normal	/	0.95
	KNN	Pearson	Square	/	0.81	Pearson	Square	/	0.92	Spearman	Normal	/	0.92	Spearman	Normal	/	0.93	Pearson	Normal	/	0.92
	Decision Tree	Pearson	Square	/	0.81	Pearson	Square	/	0.89	Pearson	Normal	/	0.89	Pearson	Square	/	0.89	Pearson	Normal	/	0.89
ORL	LinearSVM	Pearson	Normal	/	0.50	Spearman	Normal	/	0.84	Spearman	Normal	/	0.89	Spearman	Normal	/	0.88	Spearman	Normal	/	0.89
	KNN	Spearman	Square	/	0.43	Spearman	Square	/	0.57	Energy	Normal	0.50	0.64	Energy	Normal	0.20	0.66	Energy	Normal	0.30	0.68
	Decision Tree	Pearson	Normal	/	0.40	Spearman	Square	/	0.45	Pearson	Normal	/	0.48	Pearson	Normal	/	0.51	Pearson	Normal	/	0.53
warpAR10P	LinearSVM	Energy	Normal	0.60	0.46	Energy	Normal	0.40	0.74	Energy	Normal	0.70	0.85	Pearson	Square	/	0.86	Energy	Normal	0.60	0.89
	KNN	Energy	Normal	0.70	0.38	Energy	Normal	0.10	0.42	Energy	Normal	0.10	0.45	Energy	Normal	0.20	0.49	Energy	Normal	0.10	0.47
	Decision Tree	Energy	Normal	0.50	0.46	Energy	Normal	0.10	0.70	Energy	Normal	0.10	0.61	Energy	Normal	0.10	0.64	Energy	Normal	0.60	0.65
warpPIE10P	LinearSVM	Energy	Normal	0.20	0.78	Energy	Normal	0.20	0.98	Energy	Normal	0.10	0.99	Energy	Normal	0.20	1.00	Energy	Normal	0.10	1.00
	KNN	Energy	Normal	0.20	0.71	Energy	Normal	0.40	0.86	Energy	Normal	0.10	0.86	Energy	Normal	0.20	0.87	Energy	Normal	0.10	0.87
	Decision Tree	Energy	Normal	0.10	0.66	Energy	Normal	0.30	0.75	Energy	Normal	0.40	0.81	Energy	Normal	0.70	0.84	Energy	Normal	0.10	0.80
Yale	LinearSVM	Spearman	Normal	/	0.49	Pearson	Normal	/	0.63	Pearson	Normal	/	0.69	Pearson	Square	/	0.73	Energy	Normal	0.20	0.74
	KNN	Spearman	Normal	/	0.43	Pearson	Square	/	0.56	Spearman	Normal	/	0.56	Pearson	Normal	/	0.57	Spearman	Square	/	0.60
	Decision Tree	Pearson	Square	/	0.38	Pearson	Square	/	0.49	Pearson	Normal	/	0.48	Spearman	Normal	/	0.56	Pearson	Normal	/	0.52
USPS	LinearSVM	Energy	Normal	0.70	0.71	Energy	Normal	0.20	0.90	Energy	Normal	0.10	0.92	Pearson	Normal	/	0.92	Pearson	Normal	/	0.92
	KNN	Energy	Normal	0.50	0.77	Energy	Normal	0.90	0.94	Energy	Normal	0.10	0.94	Energy	Normal	0.30	0.95	Spearman	Normal	/	0.95
	Decision Tree	Spearman	Square	/	0.71	Energy	Normal	0.90	0.84	Energy	Normal	0.70	0.85	Pearson	Normal	/	0.85	Energy	Normal	0.30	0.86
colon	LinearSVM	Energy	Normal	0.30	0.73	Energy	Normal	0.80	0.71	Energy	Normal	0.90	0.74	Energy	Normal	0.90	0.79	Energy	Normal	0.60	0.77
	KNN	Energy	Normal	0.90	0.83	Energy	Normal	0.30	0.76	Energy	Normal	0.30	0.80	Energy	Normal	0.90	0.76	Energy	Normal	0.90	0.82
	Decision Tree	Pearson	Normal	/	0.67	Energy	Normal	0.90	0.70	Energy	Normal	0.60	0.76	Energy	Normal	0.50	0.71	Energy	Normal	0.50	0.76
GLIOMA	LinearSVM	Pearson	Square	/	0.65	Pearson	Normal	/	0.69	Energy	Normal	0.20	0.71	Energy	Normal	0.50	0.76	Pearson	Square	/	0.72
	KNN	Pearson	Normal	/	0.58	Energy	Normal	0.80	0.72	Pearson	Normal	/	0.68	Pearson	Normal	/	0.77	Pearson	Normal	/	0.71
	Decision Tree	Pearson	Square	/	0.60	Energy	Normal	0.70	0.72	Energy	Normal	0.40	0.75	Energy	Normal	0.40	0.69	Energy	Normal	0.70	0.62
lung	LinearSVM	Spearman	Square	/	0.74	Pearson	Normal	/	0.96	Pearson	Normal	/	0.97	Spearman	Normal	/	0.97	Energy	Normal	0.90	0.94
	KNN	Spearman	Square	/	0.61	Pearson	Square	/	0.79	Energy	Normal	0.40	0.76	Energy	Normal	0.60	0.79	Spearman	Square	/	0.80
	Decision Tree	Pearson	Square	/	0.75	Spearman	Normal	/	0.87	Pearson	Normal	/	0.87	Pearson	Normal	/	0.79	Pearson	Normal	/	0.90
lung_small	LinearSVM	Energy	Normal	0.80	0.54	Energy	Normal	0.80	0.72	Pearson	Square	/	0.76	Pearson	Normal	/	0.75	Spearman	Square	/	0.79
	KNN	Energy	Normal	0.60	0.59	Energy	Normal	0.60	0.77	Energy	Normal	0.10	0.79	Energy	Normal	0.40	0.76	Energy	Normal	0.20	0.71
	Decision Tree	Spearman	Normal	/	0.48	Energy	Normal	0.80	0.57	Pearson	Normal	/	0.58	Energy	Normal	0.80	0.56	Energy	Normal	0.10	0.68
lymphoma	LinearSVM	Pearson	Square	/	0.47	Energy	Normal	0.70	0.58	Energy	Normal	0.60	0.66	Energy	Normal	0.20	0.75	Energy	Normal	0.50	0.79
	KNN	Pearson	Square	/	0.41	Spearman	Square	/	0.54	Energy	Normal	0.70	0.61	Energy	Normal	0.70	0.67	Energy	Normal	0.70	0.63
	Decision Tree	Pearson	Square	/	0.44	Energy	Normal	0.20	0.57	Energy	Normal	0.30	0.50	Pearson	Normal	/	0.57	Energy	Normal	0.20	0.56
GISETTE	LinearSVM	Energy	Normal	0.20	0.66	Energy	Normal	0.40	0.81	Energy	Normal	0.40	0.87	Energy	Normal	0.40	0.90	Energy	Normal	0.40	0.91
	KNN	Energy	Normal	0.20	0.59	Energy	Normal	0.90	0.77	Energy	Normal	0.40	0.80	Energy	Normal	0.40	0.84	Energy	Normal	0.40	0.84
	Decision Tree	Energy	Normal	0.20	0.61	Energy	Normal	0.90	0.79	Energy	Normal	0.40	0.85	Energy	Normal	0.40	0.89	Energy	Normal	0.40	0.89
Isolet	LinearSVM	Spearman	Square	/	0.51	Pearson	Normal	/	0.74	Pearson	Normal	/	0.81	Pearson	Square	/	0.88	Pearson	Normal	/	0.89
	KNN	Pearson	Square	/	0.51	Pearson	Normal	/	0.69	Spearman	Normal	/	0.74	Spearman	Normal	/	0.79	Spearman	Normal	/	0.81
	Decision Tree	Pearson	Normal	/	0.46	Pearson	Normal	/	0.65	Pearson	Normal	/	0.70	Pearson	Normal	/	0.73	Pearson	Square	/	0.75
MADELON	LinearSVM	Pearson	Square	/	0.61	Energy	Normal	0.30	0.60	Pearson	Square	/	0.58	Spearman	Square	/	0.57	Energy	Normal	0.30	0.57
	KNN	Spearman	Normal	/	0.75	Pearson	Square	/	0.72	Spearman	Square	/	0.66	Spearman	Square	/	0.62	Pearson	Normal	/	0.59
	Decision Tree	Pearson	Normal	/	0.72	Spearman	Normal	/	0.79	Spearman	Normal	/	0.76	Spearman	Normal	/	0.77	Spearman	Normal	/	0.75

F. Additional out-of-sample evaluations

To prevent from obtaining metric-dependent out-of-sample results, in addition to the Balanced Accuracy score (see Equation 5), we consider two additional metrics: (i) the F1 score (ii) and the Matthews Correlation Coefficient (Matthews, 1975).

The general formulation for the F1 score is

$$F1 = 2 \times \frac{\frac{1}{|Z|}\sum_{z\in Z}\left(\frac{\text{TP}_z}{\text{TP}_z+\text{FP}_z}\right) \times \frac{1}{|Z|}\sum_{z\in Z}\left(\frac{\text{TP}_z}{\text{TP}_z+\text{FN}_z}\right)}{\frac{1}{|Z|}\sum_{z\in Z}\left(\frac{\text{TP}_z}{\text{TP}_z+\text{FP}_z}\right) + \frac{1}{|Z|}\sum_{z\in Z}\left(\frac{\text{TP}_z}{\text{TP}_z+\text{FN}_z}\right)}. \tag{8}$$

TP is the number of outcomes where the model correctly classifies a sample as belonging to a positive class (or detects an event of interest), when in fact it does belong to that class (or the event is present). FP is the number of outcomes where the model incorrectly classifies a sample as belonging to a positive class (or detects an event of interest), when in fact it does not belong to that class (or the event is not present). FN is the number of outcomes where the model incorrectly classifies a sample as belonging to a negative class (or fails to detect an event of interest), when in fact it belongs to a positive class (or the event of interest is present). $|Z|$ indicates the cardinality of the set of different classes.

The general formulation for the MCC is

$$MCC = \frac{(C \times S) - (T \cdot P)}{\sqrt{S^2 - (P \cdot P)} \times \sqrt{S^2 - (T \cdot T)}} \tag{9}$$

where T is a vector containing the number of times each class $z \in Z$ truly occurs, P is a vector containing the number of times each class $z \in Z$ is predicted, C is the total number of samples correctly predicted and S is the total number of samples. Given Equations 8 and 9, it is easy for the interested reader to reconstruct the formulation for the binary case.

For each performance metric, we use the implementation provided by the 'scikit-learn' Python package (Pedregosa et al., 2011) at the following links:

- Balanced Accuracy score: https://github.com/scikit-learn/scikit-learn/blob/98cf537f5/sklearn/metrics/_classification.py#L2111
- F1 score: https://github.com/scikit-learn/scikit-learn/blob/98cf537f5/sklearn/metrics/_classification.py#L1011
- Matthews Correlation Coefficient: https://github.com/scikit-learn/scikit-learn/blob/98cf537f5/sklearn/metrics/_classification.py#L848

Table 8. Subset size-dependent, out-of-sample results using a LinearSVM classifier. We use three evaluation metrics: balanced accuracy score, F1 score and Matthews Correlation Coefficient. For each dataset, we boldly highlight the combination between feature selection schema and classifier producing the best out-of-sample result.

		Linear SVM									
		10		**50**		**100**		**150**		**200**	
		Inf-FS$_U$	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**
PCMAC	Balanced Accuracy	0.52	0.50	0.57	0.67	0.59	0.70	0.61	**0.71**	0.62	0.69
	F1 Score	0.52	0.35	0.57	0.67	0.59	0.70	0.61	**0.71**	0.62	0.69
	MCC	0.05	-0.02	0.13	0.34	0.19	0.41	0.23	**0.44**	0.25	0.39
RELATHE	Balanced Accuracy	0.47	0.49	0.43	**0.53**	0.51	0.53	0.44	0.49	0.53	0.53
	F1 Score	0.33	0.37	0.40	0.48	0.50	0.51	0.43	0.48	0.50	**0.53**
	MCC	-0.14	-0.06	-0.15	0.06	0.01	**0.07**	-0.12	-0.01	0.07	0.07
COIL20	Balanced Accuracy	0.52	0.63	0.77	0.90	0.84	0.92	0.90	0.94	0.94	**0.96**
	F1 Score	0.44	0.58	0.76	0.90	0.83	0.92	0.89	0.94	0.94	**0.95**
	MCC	0.50	0.62	0.76	0.90	0.83	0.92	0.89	0.94	0.94	**0.95**
ORL	Balanced Accuracy	0.40	0.44	0.63	0.88	0.72	0.89	0.86	0.93	0.84	**0.94**
	F1 Score	0.33	0.39	0.61	0.86	0.71	0.88	0.85	**0.93**	0.84	0.93
	MCC	0.39	0.43	0.63	0.87	0.71	0.89	0.86	0.93	0.84	**0.94**
warpAR10P	Balanced Accuracy	0.33	0.44	0.56	0.78	0.72	0.85	0.70	**0.95**	0.75	0.85
	F1 Score	0.29	0.43	0.56	0.76	0.71	0.85	0.69	**0.94**	0.74	0.84
	MCC	0.27	0.38	0.52	0.75	0.69	0.83	0.67	**0.95**	0.72	0.84
warpPIE10P	Balanced Accuracy	0.85	0.89	0.95	**1.00**	0.98	1.00	1.00	1.00	1.00	1.00
	F1 Score	0.85	0.89	0.95	**1.00**	0.98	1.00	1.00	1.00	1.00	1.00
	MCC	0.85	0.88	0.95	**1.00**	0.98	1.00	1.00	1.00	1.00	1.00
Yale	Balanced Accuracy	0.14	0.33	0.25	0.50	0.39	0.67	0.37	0.69	0.53	**0.70**
	F1 Score	0.12	0.31	0.25	0.50	0.38	0.67	0.36	**0.70**	0.53	0.66
	MCC	0.08	0.27	0.21	0.47	0.36	0.66	0.34	**0.68**	0.51	0.68
USPS	Balanced Accuracy	0.72	0.65	0.90	0.90	0.91	0.92	0.92	**0.93**	0.92	0.93
	F1 Score	0.71	0.64	0.90	0.91	0.91	0.92	0.92	**0.93**	0.92	0.93
	MCC	0.72	0.65	0.90	0.90	0.91	0.92	0.92	**0.93**	0.92	0.93
colon	Balanced Accuracy	0.70	0.69	0.69	0.66	**0.92**	0.82	0.85	0.74	0.85	0.88
	F1 Score	0.71	0.68	0.68	0.66	**0.89**	0.82	0.83	0.76	0.83	0.84
	MCC	0.42	0.37	0.37	0.32	**0.81**	0.65	0.67	0.53	0.67	0.72
GLIOMA	Balanced Accuracy	**0.61**	0.25	0.30	0.30	0.30	0.38	0.60	0.41	0.59	0.25
	F1 Score	0.57	0.12	0.24	0.28	0.26	0.28	**0.59**	0.35	0.58	0.13
	MCC	**0.50**	0.00	0.03	0.06	0.10	0.22	0.47	0.26	0.46	-0.03
lung	Balanced Accuracy	0.39	0.47	0.67	0.89	0.81	**0.95**	0.71	0.87	0.90	0.81
	F1 Score	0.42	0.50	0.67	0.88	0.79	**0.91**	0.68	0.86	0.87	0.83
	MCC	0.31	0.61	0.70	0.80	0.77	**0.85**	0.77	0.77	0.79	0.80
lung_small	Balanced Accuracy	0.49	0.57	0.76	0.79	0.82	0.68	0.79	0.75	0.82	**0.93**
	F1 Score	0.52	0.55	0.73	0.71	0.78	0.65	0.74	0.69	0.78	**0.93**
	MCC	0.51	0.67	0.72	0.78	0.84	0.73	0.79	0.74	0.84	**0.90**
lymphoma	Balanced Accuracy	0.22	0.50	0.58	0.96	0.78	0.87	0.90	0.82	0.81	**0.98**
	F1 Score	0.22	0.47	0.52	0.91	0.77	0.83	0.92	0.78	0.77	**0.96**
	MCC	0.33	0.66	0.44	0.84	0.81	**0.91**	0.91	0.82	0.78	0.91
GISETTE	Balanced Accuracy	0.50	0.49	0.48	0.47	0.51	**0.52**	0.47	0.50	0.49	0.50
	F1 Score	0.38	0.49	0.46	0.44	0.46	**0.52**	0.43	0.45	0.45	0.46
	MCC	-0.01	-0.01	-0.05	-0.07	0.02	**0.05**	-0.08	0.00	-0.02	0.00
Isolet	Balanced Accuracy	0.32	0.51	0.74	0.78	0.81	0.82	0.88	0.83	0.89	**0.89**
	F1 Score	0.26	0.48	0.73	0.78	0.81	0.82	0.88	0.83	0.89	**0.89**
	MCC	0.30	0.49	0.73	0.77	0.80	0.81	0.88	0.82	0.88	**0.89**
MADELON	Balanced Accuracy	0.59	**0.59**	0.58	0.56	0.55	0.57	0.54	0.57	0.57	0.57
	F1 Score	0.59	**0.59**	0.58	0.55	0.55	0.57	0.54	0.57	0.57	0.57
	MCC	0.18	**0.18**	0.16	0.11	0.10	0.14	0.09	0.15	0.14	0.14

Table 9. Subset size-dependent, out-of-sample results using a KNN classifier. We use three evaluation metrics: the Balanced Accuracy score, the F1 score and the Matthews Correlation Coefficient. For each dataset, we boldly highlight the combination between feature selection schema and classifier producing the best out-of-sample result.

		KNN									
		10		**50**		**100**		**150**		**200**	
		Inf-FS$_U$	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**
PCMAC	Balanced Accuracy	0.52	0.53	0.57	0.61	0.61	**0.62**	0.61	0.62	0.61	0.62
	F1 Score	0.52	0.41	0.57	0.61	0.61	0.61	0.59	**0.63**	0.60	0.62
	MCC	0.05	0.15	0.15	0.22	0.23	0.25	0.18	**0.26**	0.21	0.24
RELATHE	Balanced Accuracy	0.46	0.46	0.50	**0.57**	0.48	0.49	0.48	0.49	0.48	0.49
	F1 Score	0.37	0.37	0.47	**0.54**	0.46	0.46	0.45	0.45	0.48	0.46
	MCC	-0.10	-0.10	0.00	**0.17**	-0.04	-0.03	-0.09	-0.10	-0.03	-0.05
COIL20	Balanced Accuracy	0.70	0.82	0.86	**0.93**	0.93	0.93	0.93	0.93	0.93	0.93
	F1 Score	0.67	0.81	0.85	0.93	0.92	0.93	0.96	0.94	**0.97**	0.93
	MCC	0.69	0.81	0.85	0.93	0.92	0.93	0.95	0.94	**0.97**	0.93
ORL	Balanced Accuracy	0.38	0.52	0.52	**0.77**	0.62	0.70	0.62	0.70	0.62	0.70
	F1 Score	0.38	0.49	0.49	0.75	0.61	0.68	0.73	0.69	0.69	**0.76**
	MCC	0.37	0.52	0.52	0.76	0.62	0.69	0.73	0.70	0.72	**0.77**
warpAR10P	Balanced Accuracy	0.36	0.30	0.36	**0.51**	0.43	0.46	0.43	0.46	0.43	0.46
	F1 Score	0.32	0.26	0.32	**0.50**	0.41	0.48	0.27	0.37	0.37	0.47
	MCC	0.29	0.24	0.30	**0.47**	0.38	0.41	0.25	0.32	0.37	0.44
warpPIE10P	Balanced Accuracy	0.83	0.72	0.86	0.91	0.92	**0.97**	0.92	0.97	0.92	0.97
	F1 Score	0.83	0.69	0.86	0.91	0.92	**0.97**	0.89	0.92	0.89	0.95
	MCC	0.81	0.69	0.84	0.90	0.91	**0.97**	0.88	0.91	0.88	0.95
Yale	Balanced Accuracy	0.14	**0.42**	0.28	0.41	0.26	0.42	0.26	0.42	0.26	0.42
	F1 Score	0.12	0.40	0.27	0.40	0.23	0.43	0.44	0.38	0.32	**0.49**
	MCC	0.08	0.39	0.23	0.36	0.23	0.39	0.41	0.34	0.33	**0.48**
USPS	Balanced Accuracy	0.78	0.77	0.94	0.94	**0.96**	0.95	0.96	0.95	0.96	0.95
	F1 Score	0.78	0.77	0.94	0.94	**0.96**	0.95	0.96	0.95	0.95	0.95
	MCC	0.78	0.78	0.94	0.94	**0.96**	0.95	0.96	0.95	0.95	0.95
colon	Balanced Accuracy	0.77	0.82	**0.89**	0.77	0.89	0.85	0.89	0.85	0.89	0.85
	F1 Score	0.77	0.82	0.89	0.77	0.89	0.83	**1.00**	0.71	0.83	0.77
	MCC	0.55	0.65	0.77	0.55	0.77	0.67	**1.00**	0.42	0.67	0.55
GLIOMA	Balanced Accuracy	0.24	0.10	0.40	0.40	0.42	**0.62**	0.42	0.62	0.42	0.62
	F1 Score	0.19	0.06	0.39	0.40	0.40	0.49	**0.52**	0.49	0.50	0.49
	MCC	-0.04	-0.31	0.26	0.27	0.30	**0.48**	0.46	0.48	0.46	0.48
lung	Balanced Accuracy	0.33	0.51	0.65	**0.79**	0.71	0.65	0.71	0.65	0.71	0.65
	F1 Score	0.35	0.52	0.65	**0.84**	0.70	0.67	0.71	0.69	0.81	0.84
	MCC	0.24	0.56	0.70	**0.83**	0.77	0.72	0.80	0.76	0.77	0.83
lung_small	Balanced Accuracy	0.57	0.61	0.80	0.87	0.82	**0.90**	0.82	0.90	0.82	0.90
	F1 Score	0.55	0.57	0.77	0.87	0.80	0.85	**0.89**	0.85	0.85	0.72
	MCC	0.51	0.64	0.78	0.84	0.84	0.84	**0.90**	0.84	0.84	0.73
lymphoma	Balanced Accuracy	0.44	0.50	0.60	**0.74**	0.69	0.69	0.69	0.69	0.69	0.69
	F1 Score	0.45	0.49	0.61	0.70	0.67	0.65	**0.75**	0.72	0.65	0.70
	MCC	0.49	0.66	0.76	0.86	0.86	0.86	0.81	**0.91**	0.86	0.86
GISETTE	Balanced Accuracy	0.49	0.51	0.52	**0.54**	0.50	0.51	0.50	0.51	0.50	0.51
	F1 Score	0.38	0.50	0.49	**0.52**	0.40	0.43	0.34	0.48	0.34	0.47
	MCC	-0.03	0.02	0.04	**0.09**	-0.01	0.03	-0.03	0.08	-0.03	-0.01
Isolet	Balanced Accuracy	0.32	0.49	0.72	0.73	0.78	**0.78**	0.78	0.78	0.78	0.78
	F1 Score	0.31	0.48	0.72	0.73	0.78	0.77	**0.83**	0.81	0.82	0.83
	MCC	0.29	0.47	0.71	0.72	0.78	0.77	0.82	0.81	0.81	**0.83**
MADELON	Balanced Accuracy	0.61	**0.78**	0.58	0.74	0.64	0.66	0.64	0.66	0.64	0.66
	F1 Score	0.61	**0.78**	0.58	0.74	0.64	0.66	0.62	0.64	0.57	0.62
	MCC	0.23	**0.56**	0.16	0.48	0.29	0.32	0.24	0.29	0.14	0.26

Table 10. Subset size-dependent, out-of-sample results using a Decision Tree classifier. We use three evaluation metrics: the Balanced Accuracy score, the F1 score and the Matthews Correlation Coefficient. For each dataset, we boldly highlight the combination between feature selection schema and classifier producing the best out-of-sample result.

		Decision Tree									
		10		**50**		**100**		**150**		**200**	
		Inf-FS$_U$	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**	**Inf-FS$_U$**	**TFS**
PCMAC	Balanced Accuracy	0.53	0.50	0.56	0.69	0.58	**0.71**	0.58	0.71	0.58	0.71
	F1 Score	0.53	0.35	0.56	0.69	0.58	0.71	0.57	0.68	0.60	**0.73**
	MCC	0.06	-0.01	0.13	0.38	0.17	0.42	0.14	0.36	0.21	**0.46**
RELATHE	Balanced Accuracy	0.49	0.50	0.51	**0.51**	0.49	0.42	0.49	0.42	0.49	0.42
	F1 Score	0.35	0.34	0.41	0.47	0.46	0.41	0.45	**0.48**	0.48	0.44
	MCC	-0.02	-0.02	**0.03**	0.02	-0.01	-0.18	-0.04	0.02	-0.04	0.03
COIL20	Balanced Accuracy	0.68	0.81	0.83	0.89	0.85	**0.90**	0.85	0.90	0.85	0.90
	F1 Score	0.67	0.81	0.83	0.89	0.85	**0.90**	0.89	0.90	0.90	0.90
	MCC	0.67	0.80	0.82	0.89	0.84	**0.90**	0.89	0.90	0.90	0.90
ORL	Balanced Accuracy	0.36	0.39	0.42	0.48	0.49	**0.54**	0.49	0.54	0.49	0.54
	F1 Score	0.33	0.37	0.40	0.46	0.49	0.52	0.57	0.58	0.49	**0.60**
	MCC	0.34	0.38	0.41	0.47	0.48	0.53	0.58	0.60	0.48	**0.61**
warpAR10P	Balanced Accuracy	0.37	0.33	0.46	**0.59**	0.55	0.59	0.55	0.59	0.55	0.59
	F1 Score	0.35	0.33	0.44	0.60	0.52	0.61	0.41	0.64	0.67	**0.81**
	MCC	0.29	0.27	0.41	0.55	0.49	0.55	0.35	0.61	0.63	**0.78**
warpPIE10P	Balanced Accuracy	0.74	0.74	0.80	0.73	0.77	**0.85**	0.77	0.85	0.77	0.85
	F1 Score	0.74	0.75	0.78	0.73	0.78	0.85	0.75	**0.87**	0.76	0.79
	MCC	0.72	0.72	0.78	0.71	0.76	0.84	0.74	**0.86**	0.74	0.79
Yale	Balanced Accuracy	0.17	0.31	0.26	0.34	0.39	**0.42**	0.39	0.42	0.39	0.42
	F1 Score	0.15	0.31	0.25	0.34	0.39	0.41	0.47	0.43	0.45	**0.53**
	MCC	0.12	0.26	0.21	0.30	0.36	0.38	**0.49**	0.38	0.40	0.49
USPS	Balanced Accuracy	0.73	0.72	0.84	0.85	0.85	**0.86**	0.85	0.86	0.85	0.86
	F1 Score	0.73	0.72	0.84	0.85	0.85	0.86	0.86	0.86	**0.88**	0.87
	MCC	0.73	0.72	0.84	0.85	0.85	0.86	0.86	0.86	**0.88**	0.87
colon	Balanced Accuracy	0.61	0.64	0.82	0.74	0.76	**0.92**	0.76	0.92	0.76	0.92
	F1 Score	0.58	0.64	0.82	0.76	0.73	**0.89**	0.89	0.79	0.83	0.79
	MCC	0.21	0.45	0.65	0.53	0.51	**0.81**	0.77	0.65	0.67	0.65
GLIOMA	Balanced Accuracy	0.34	**0.61**	0.36	0.31	0.35	0.44	0.35	0.44	0.35	0.44
	F1 Score	0.37	**0.54**	0.25	0.18	0.34	0.39	0.22	0.22	0.19	0.33
	MCC	0.19	**0.41**	0.06	0.10	0.19	0.21	0.19	0.14	0.08	0.33
lung	Balanced Accuracy	0.44	0.70	0.75	0.71	**0.87**	0.70	0.87	0.70	0.87	0.70
	F1 Score	0.44	0.70	0.67	0.75	0.83	0.66	**0.88**	0.73	0.75	0.75
	MCC	0.39	0.64	0.64	0.69	0.78	0.69	**0.79**	0.71	0.67	0.70
lung_small	Balanced Accuracy	0.46	0.42	0.58	**0.63**	0.47	0.57	0.47	0.57	0.47	0.57
	F1 Score	0.42	0.37	0.47	0.58	0.42	0.48	0.40	**0.63**	0.52	0.37
	MCC	0.45	0.50	**0.58**	0.57	0.40	0.47	0.47	0.55	0.44	0.41
lymphoma	Balanced Accuracy	0.20	**0.69**	0.45	0.55	0.45	0.44	0.45	0.44	0.45	0.44
	F1 Score	0.17	**0.67**	0.37	0.45	0.38	0.39	0.59	0.49	0.46	0.49
	MCC	0.18	**0.64**	0.50	0.57	0.44	0.50	0.62	0.59	0.57	0.60
GISETTE	Balanced Accuracy	**0.52**	0.50	0.44	0.52	0.48	0.47	0.48	0.47	0.48	0.47
	F1 Score	0.45	**0.50**	0.41	0.49	0.48	0.46	0.46	0.48	0.49	0.44
	MCC	**0.05**	0.00	-0.14	0.05	-0.04	-0.07	-0.01	-0.01	-0.01	-0.04
Isolet	Balanced Accuracy	0.27	0.43	0.69	0.67	**0.73**	0.71	0.73	0.71	0.73	0.71
	F1 Score	0.28	0.43	0.68	0.67	0.73	0.70	0.74	0.72	**0.78**	0.72
	MCC	0.24	0.41	0.67	0.66	0.72	0.69	0.73	0.72	**0.77**	0.72
MADELON	Balanced Accuracy	0.58	0.66	0.70	**0.81**	0.78	0.79	0.78	0.79	0.78	0.79
	F1 Score	0.58	0.66	0.70	**0.81**	0.78	0.79	0.75	0.77	0.73	0.77
	MCC	0.16	0.31	0.40	**0.62**	0.55	0.57	0.50	0.54	0.47	0.53

G. Statistical Validation

The statistical significance of results discussed in Section 3 is assessed in Table 11. Here we report the p-values obtained performing a 15×2cv t-test as described in Section 2.5. Specifically, p-values > 0.1 are reported in their numerical form, p-values ≤ 0.1 and > 0.05 are marked as *, p-values ≤ 0.05 and > 0.01 are marked as **, p-values ≤ 0.01 and > 0.001 are marked as *** and p-values ≤ 0.001 are marked as ****.

Table 11. Comparison between Inf-FS$_U$' and TFS' p-values obtained performing a 15×2 cv paired t-test. p-values > 0.1 are reported in their numerical form, p-values ≤ 0.1 and > 0.05 are marked as *, p-values ≤ 0.05 and > 0.01 are marked as **, p-values ≤ 0.01 and > 0.001 are marked as *** and p-values ≤ 0.001 are marked as ****. $(\vee)$ and $(\wedge)$ symbols indicate that, when the two feature selection schemes combined with the same classifier, produce statistically robust different results, TFS performs, respectively, better or worse than Inf-FS$_U$ according to results reported in Table 2.

	LinearSVM					**KNN**					**DT**				
	10	**50**	**100**	**150**	**200**	**10**	**50**	**100**	**150**	**200**	**10**	**50**	**100**	**150**	**200**
PCMAC	$^{*(\wedge)}$	0.50	0.79	0.80	0.49	0.19	0.37	0.90	0.81	0.94	$^{*(\wedge)}$	0.69	0.93	0.75	0.73
RELATHE	0.87	$^{**(\vee)}$	$^{****(\vee)}$	$^{***(\vee)}$	$^{**(\vee)}$	0.61	0.17	$^{****(\vee)}$	$^{*(\wedge)}$	0.14	0.90	$^{***(\vee)}$	0.58	$^{*(\vee)}$	0.47
COIL20	$^{***(\vee)}$	$^{****(\vee)}$	$^{****(\vee)}$	$^{***(\vee)}$	$^{**(\vee)}$	$^{**(\vee)}$	$^{**(\vee)}$	$^{**(\vee)}$	0.39	0.40	$^{*(\vee)}$	$^{**(\vee)}$	$^{***(\vee)}$	$^{***(\vee)}$	$^{**(\vee)}$
ORL	0.14	$^{***(\vee)}$	0.22	$^{**(\vee)}$	$^{***(\vee)}$	$^{**(\vee)}$	$^{***(\vee)}$	0.72	0.44	0.51	0.88	0.49	0.67	$^{**(\vee)}$	0.93
warpAR10P	0.23	$^{***(\vee)}$	0.38	0.23	0.20	$^{**(\wedge)}$	$^{**(\vee)}$	$^{**(\vee)}$	0.48	0.69	0.77	$^{***(\vee)}$	0.32	0.16	0.39
warpPIE10P	0.98	0.52	0.67	0.39	1.00	0.50	0.93	0.17	1.00	0.51	0.41	1.00	0.65	0.39	0.68
Yale	$^{***(\vee)}$	$^{**(\vee)}$	0.17	0.19	0.64	0.81	$^{**(\vee)}$	0.28	0.78	0.10	0.60	$^{**(\vee)}$	0.90	0.45	0.23
USPS	0.89	0.38	0.29	$^{***(\vee)}$	$^{*(\vee)}$	0.69	0.19	$^{***(\wedge)}$	$^{*(\wedge)}$	0.26	$^{*(\wedge)}$	0.74	0.71	0.62	0.95
colon	0.15	0.42	0.19	0.37	0.53	0.92	0.48	0.63	1.00	0.18	0.89	0.35	0.15	0.59	0.50
GLIOMA	0.54	0.63	0.45	$^{*(\wedge)}$	0.74	0.68	0.73	0.25	0.65	0.91	$^{**(\vee)}$	0.52	$^{*(\vee)}$	$^{*(\wedge)}$	$^{*(\vee)}$
lung	$^{***(\vee)}$	0.12	$^{***(\vee)}$	$^{**(\vee)}$	0.19	$^{***(\vee)}$	0.46	0.61	0.84	0.58	$^{***(\vee)}$	0.11	0.28	0.47	0.86
lung_small	0.72	0.83	$^{**(\wedge)}$	0.54	0.32	0.38	0.77	0.32	0.82	0.72	0.73	0.99	0.53	$^{*(\vee)}$	0.63
lymphoma	0.56	0.26	0.37	0.74	0.92	1.00	0.69	0.41	0.25	0.86	0.35	0.58	0.32	0.45	0.85
GISETTE	$^{****(\wedge)}$	$^{***(\wedge)}$	$^{***(\vee)}$	$^{***(\vee)}$	$^{**(\vee)}$	$^{****(\vee)}$	$^{****(\vee)}$	$^{****(\vee)}$	$^{****(\vee)}$	$^{****(\wedge)}$	$^{****(\wedge)}$	$^{***(\vee)}$	$^{***(\wedge)}$	$^{**(\wedge)}$	$^{*(\vee)}$
Isolet	$^{****(\vee)}$	$^{***(\vee)}$	0.15	0.15	$^{**(\vee)}$	$^{****(\vee)}$	1.00	0.28	0.38	0.88	$^{**(\vee)}$	0.81	0.61	$^{*(\wedge)}$	0.39
MADELON	$^{*(\vee)}$	0.75	0.93	0.19	0.15	0.91	0.69	0.22	0.65	0.49	0.32	0.65	0.57	0.69	0.59