(Un)supervised Univariate Feature Extraction and Selection for Dimensional Data

Patrik Cavina[1,*,†], Federico Manzella[1,*,†], Giovanni Pagliarini[1,*,†], Guido Sciavicco[1,*,†] and Eduard I. Stan[1,2,*,†]

[1]*ACLAI Laboratory, University of Ferrara (Italy)*
[2]*DBS Group, Free University of Bozen-Bolzano (Italy)*

Abstract

Feature selection, defined as the automatic selection of the most relevant features from a machine learning dataset, has three objectives: to improve the prediction performance of the predictors, to provide faster and more cost-effective predictors, and to offer a better understanding of the underlying process that generated the data. In the case of dimensional (e.g., temporal or spatial) data, feature selection is often approached using standard methods that neglect or denature the dimensional component. This paper provides a first step towards systematic and general dimensional feature selection, with a portfolio of supervised and unsupervised, filter-based selectors that can be naturally combined into an end-to-end methodology. In a hypothesis-testing setting, our experiments show that our approach can extract provably relevant features in both the temporal and spatial cases.

Keywords

Feature selection, Filter-based selection, Non-tabular data, Dimensional data

1. Introduction

Data analysis involves inspecting, cleaning, transforming, and conceptualizing data. The goal is simple: uncover valuable insights, foster informed conclusions, and provide strategic decision-making. This practice unfolds through multiple layers of a data pre-processing pipeline, including *feature extraction* and *selection*. Feature extraction is the process of producing meaningful features that capture the underlying complexity of data, revealing valuable insights that may be hidden in the raw data by enhancing its predictive power. Feature selection, on the other hand, identifies and selects the most informative subset of features, with the objective of reducing the dimensionality of the data and eliminating noise and redundancy while preserving as much information as possible. Sometimes, feature engineering and selection can be combined into a sequential process that first creates several new features and then selects the most informative ones from the resulting extended set.

ITADATA2023: The 2nd Italian Conference on Big Data and Data Science, September 11–13, 2023, Naples, Italy
*Corresponding author.
†These authors contributed equally.
patrik.cavina@edu.unife.it (P. Cavina); federic.manzella@edu.unife.it (F. Manzella); giovanni.pagliarini@unife.it (G. Pagliarini); guido.sciavicco@unife.it (G. Sciavicco); ioneleduard.stan@unibz.it (E. I. Stan)
0009-0003-9085-3815 (P. Cavina); 0000-0002-4944-2163 (F. Manzella); 0000-0002-8403-3250 (G. Pagliarini); 0000-0002-9221-879X (G. Sciavicco); 0000-0001-9260-102X (E. I. Stan)

Feature selection methods can be classified into filter, wrapper, embedded and hybrid methods [1]. *Filter-based* methods are independent of the learning models and base their estimate of feature importance on heuristic or statistical ranking criteria. Filters can be characterized in two orthogonal ways: univariate versus multivariate, and supervised versus unsupervised. *Univariate* filters rank each feature independently, and thus disregard any form of inter-feature correlation. *Multivariate* filters, on the contrary, rank multiple features in batches and can account for interdependencies between features [1]. *Supervised* filters estimate the relevance of a feature (group) with respect to a target variable, that is, the label, while *unsupervised* ones select features only according to their own nature and characteristics [2]. Filters can be further characterized depending on the statistical principle on which they are based: typical choices in the unsupervised case include filters based on the *variance*, *entropy*, or *Laplacian score* [3], while supervised ones can be based on *correlation/covariance*, *entropy gain*, *supervised Laplacian score*, *hypothesis test(s)*, or *discriminating power* [4]. *Wrapper* methods, unlike filter ones, use a learning model to evaluate the performance of different feature subsets; they iterate a search process until an optimal result, or some stopping condition, is reached, where the search strategies can be random, sequential, or heuristic [5]. *Embedded* methods integrate the feature selection process into the learning model. *Hybrid* methods are combinations of filters and wrappers that take advantage of both techniques: the filter reduces the feature set to a good enough subset, and the wrapper maximizes performance. [1].

In the era of big data, more-than-tabular data embodies 95% of all existing data [6]. Tabular data is often found in spreadsheets and relational databases, which can be easily analyzed using data analytic software. Non-tabular data, such as sets of time series, images, videos, graphs and texts, is subjective and interpretative, meaning that data analytic tools may not always be readily accessible, which prompts practitioners to develop ad-hoc tools or enhance existing ones. A particular, but very common, case of non-tabular data is *d-dimensional* data, in which every instance is represented by an n-dimensional array of real functions defined on a d-dimensional discrete space, covering virtually all cases of real-world temporal ($d = 1$), spatial ($d = 2$, $d = 3$), video ($d = 3$) data, as well as scalar ones ($d = 0$).

Learning from temporal and/or spatial datasets has been primarily accomplished with *sub-symbolic* techniques, specifically neural networks; the current literature on this topic is too wide to be reviewed here. Recently, however, *symbolic* learning in both the temporal and the spatial case has started to be systematically explored. Among several examples, for the temporal case we mention [7], for extracting temporal rules, [8], for decision tree-based procedures to classify time series data, [9], in which *shapelets* are used to classify time series, and [10], in which classifiers designed to extract point-based temporal logic formulas are used to solve the classification problem involving time series. Symbolic methods, and in particular propositional decision trees, have also been applied to spatial data [11]. Recently, symbolic learning from d-dimensional data has been approached with a rather new technique known as *modal* symbolic learning. Modal symbolic learning is based on the idea that propositional logic can be replaced by modal logic as a tool to describe d-dimensional instances, and therefore, used to extract patterns. In modal symbolic learning, d-dimensional data are analyzed by considering all d-dimensional hyperintervals of a dataset as well as their qualitative relationships, and patterns are described with a suitable modal logic with enough expressive power to represent such relationships. Although the theoretical properties of modal symbolic learning techniques have

started to be studied only recently [12], modal decision trees and modal random forests have been successfully applied to several different temporal and spatial datasets [13, 14]. As it turns out, modal symbolic learning on d-dimensional data is naturally associated to feature engineering and selection specifically designed to explore d-dimensional hyperintervals. In this work, we focus on designing a univariate filter-based feature extraction and selection technique, that includes both unsupervised and supervised methods. Such methods are the d-dimensional generalization of standard ones, and can be combined into a general protocol to select the most informative combinations of features and variables, whose informative value can be assessed independently of learning models.

2. Dimensional Data

Let a *tabular instance*, also referred to as *adimensional instance*, be an object of the type $I = (v_1, \ldots, v_n)$ where $v_i \in \mathbb{R}$, for each $1 \leq i \leq n$. Each value v_i is the value for a *variable*, whose *name* is denoted by V_i; in the following, we identify variables with their names. A *tabular dataset* (or *adimensional dataset*) is, therefore, a collection $\mathscr{I} = \{I_1, \ldots, I_m\}$ of tabular instances whose values are taken from a set of *variables* $\mathscr{V} = \{V_1, \ldots, V_n\}$. Tabular datasets can be also *labelled* if each instance I is associated to a unique *label* L from a set $\mathscr{L} = \{L_1, \ldots, L_k\}$.

Tabular datasets are classic, scalar datasets as they are usually defined in data science and machine learning; note that categorical variables can always be transformed into scalar ones via one-hot encoding. Tabular datasets, however, are a particular case of dimensional ones. A *d-dimensional instance* is an n-tuple of functions $I = (v_1, \ldots, v_n)$, each defined as

$$v_i : \mathbb{N}^d \to \mathbb{R},$$

and a *d-dimensional dataset* a collection $\mathscr{I} = \{I_1, \ldots, I_m\}$ of d-dimensional instances each possibly associated to a unique label. The notion of d-dimensional dataset generalizes tabular datasets (when $d = 0$), as well as datasets of time series ($d = 1$), images ($d = 2$), and videos ($d = 3$), among others. In a d-dimensional instance too, a variable (function) v_i has a name V_i. In the following, we restrict ourselves to the case of *finite* datasets, in which we have exactly N distinct points in every dimension.

Information extraction from d-dimensional data occurs via using a predetermined set of *feature extraction functions* $\mathscr{F} = \{f_1, \ldots, f_l\}$. Each function f_i is abstractly defined as

$$f_i : \bigcup_{1 \leq j_1, \ldots, j_d \leq N} \mathbb{R}^{j_1} \times \ldots \times \mathbb{R}^{j_d} \to \mathbb{R}.$$

Examples are the *generalized mean*, *maximum* and *minimum*, but $\mathscr{F}$ may include more elaborate examples whose concrete definition may depend on the relative order of values; an example in this category in the case $d = 1$ could be the *number of local maxima* in a given interval of values. A function f is applied to a variable V to generate the *feature* $f(V)$.

There are many possible logical representations for a d-dimensional dataset. A very natural one is inspired by both the interval temporal logic $\mathscr{HS}$ [15] and the Rectangle Algebra [16], which, in turn, can be immediately generalized to the case of d-dimensions. Let $\mathbb{D}$ be an initial prefix of $\mathbb{N}$ of cardinality N, and let $\mathbb{D}^d$ its corresponding d-dimensional generalization. Let P^i_j

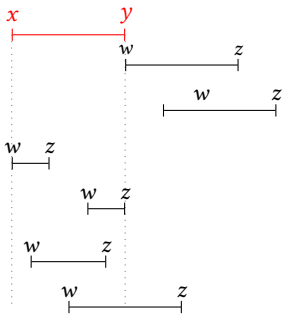
HS modality	Definition w.r.t. the interval structure			Example
$\langle A \rangle$ (after)	$[x,y]R_A[w,z]$	$\Leftrightarrow$	$y = w$	
$\langle L \rangle$ (later)	$[x,y]R_L[w,z]$	$\Leftrightarrow$	$y < w$	
$\langle B \rangle$ (begins)	$[x,y]R_B[w,z]$	$\Leftrightarrow$	$x = w \wedge z < y$	
$\langle E \rangle$ (ends)	$[x,y]R_E[w,z]$	$\Leftrightarrow$	$y = z \wedge x < w$	
$\langle D \rangle$ (during)	$[x,y]R_D[w,z]$	$\Leftrightarrow$	$x < w \wedge z < y$	
$\langle O \rangle$ (overlaps)	$[x,y]R_O[w,z]$	$\Leftrightarrow$	$x < w < y < z$	

Table 1
Allen's interval relations and HS modalities. Equality (=) is not displayed.

the $(d-1)$-dimensional discrete space passing through the j-th point of the i-th component of $\mathbb{D}^d$ and whose normal is non-null only on that component (e.g., when $d = 1$, P^1_j is simply the j-th point; when $d = 2$, P^1_j is the straight that goes through the j-th point and is parallel to the vertical axis). A *hyperinterval* H is a set

$$H = \{(P^1_{low_1}, P^1_{high_1}), \ldots, (P^d_{low_d}, P^d_{high_d})\}$$

where $low_i \leq up_i$, for each $1 \leq i \leq d$. Thus, this notion simply generalizes the one of interval in one dimension. Describing a hyperinterval is easily achieved via a d-tuple of pairs of natural numbers such as

$$H = [(x_1, y_1), \ldots, (x_d, y_d)],$$

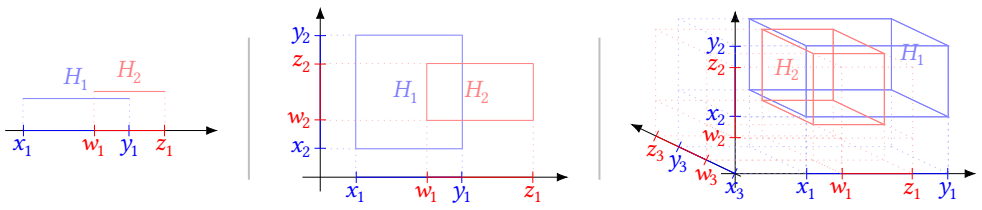
where the i-th component corresponds to the i-th dimension, and, as before, $x_i \leq y_i$, for each $1 \leq i \leq d$. A notion of a *point belonging to a hyperinterval*, denoted by $(x_1, \ldots, x_d) \in H$, emerges naturally. Given the 13 so-called *Allen's relations* that allow one to qualitatively describe the geometric arrangement between any two intervals in a linear order, whose notation and informal semantics is depicted in Tab. 1, it is immediate to generalize such relations to the case of hyperintervals. Thus, given any two hyperintervals H, H', they are related to each other by exactly one *hyperinterval relation* $R_{X_1 \ldots X_d}$, where $X_i \in \mathcal{X} = \{A, \overline{A}, L, \overline{L}, B, \overline{B}, E, \overline{E}, D, \overline{D}, O, \overline{O}, =\}$, for each $1 \leq i \leq d$, and $R_{X_1 X_2 \ldots X_d}$, which generalizes the definition of an Allen's relation to d dimensions (see Fig. 1).

Given a set $\mathcal{P}$ of propositional letters, the *logic* $\mathcal{HS}^d$ is the d-dimensional generalization of the *interval temporal logic* $\mathcal{HS}$. Well-formed $\mathcal{HS}^d$ formulas are obtained by the following syntax:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle X_1 \ldots X_d \rangle \varphi,$$

where, for each i, $X_i \in \mathcal{X}$, and $\langle X_1 \ldots X_d \rangle \neq \langle = \ldots = \rangle$. The remaining Boolean operators, as well as the universal version of each of the $13^d - 1$ existential operators can be defined as a shortcut via duality.

The strict semantics of $\mathcal{HS}^d$ is given in terms of *d-dimensional models* of the type $M = \langle \mathbb{I}(\mathbb{D}^d), \phi \rangle$, where $\mathbb{I}(\mathbb{D}^d)$ is the set of all hyperintervals over $\mathbb{D}^d$, and V is a *valuation function*



$H_1 R_O H_2$ $H_1 R_{OD} H_2$ $H_1 R_{D\bar{O}\bar{O}} H_2$

Figure 1: Examples of hyperintervals and $\mathcal{HS}^d$ relations for $d \in \{1, 2, 3\}$.

$\phi : \mathcal{P} \to 2^{\mathbb{I}(\mathbb{D}^d)}$ which assigns to every atomic proposition $p \in \mathcal{P}$ the set of hyperintervals $\phi(p)$ on which p holds. The truth of a formula φ on a given hyperinterval H in an interval model M, denoted by $M, H \Vdash \varphi$, is defined by structural induction on the complexity of formulas as follows:

$M, H \Vdash p$	if and only if	$H \in \phi(p)$, for each $p \in \mathcal{P}$;
$M, H \Vdash \neg\psi$	if and only if	$M, H \nVdash \psi$;
$M, H \Vdash \psi_1 \vee \psi_2$	if and only if	$M, H \Vdash \psi_1$ or $M, H \Vdash \psi_2$;
$M, H \Vdash \langle X_1 \dots X_d \rangle \psi$	if and only if	there exists H' s.t. $H R_{X_1 \dots X_d} H'$ and $M, H' \Vdash \psi$.

The notion of formula satisfied by a model, satisfiable formula, and valid formula are defined in the standard way.

Quite naturally, a d-dimensional instance described by the variables in $\mathcal{V}$ can be seen as a d-dimensional model, provided that, fixed a set of feature extraction functions $\mathcal{F}$, the propositional vocabulary is defined as

$$\mathcal{P} = \{f(V) \bowtie v \mid f \in \mathcal{F}, V \in \mathcal{V}, \bowtie \in \{<, \leq, =, \neq, \geq, >\}, v \in \mathbb{R}\},$$

so that, given a d-dimensional instance I, a hyperinterval $H = [(x_1, y_1), \dots, (x_d, y_d)]$, and a variable V, we can define the *value of I on V in H*, denoted $I(H, V)$ as a hypermatrix in $\mathbb{R}^{y_1 - x_1 + 1} \times \dots \times \mathbb{R}^{y_d - x_d + 1}$ whose generic element with indexes $j_1, \dots, j_d$, with $1 \leq j_i \leq y_i - x_i$, for all $[x_i, y_i] \in H$, is defined as:

$$I(H, V)^{j_1, \dots, j_d} = v(x_1 + j_1 - 1, \dots, x_d + j_d - 1).$$

Then, for a given propositional letter $p = f(V) \bowtie v$, we define

$$\phi(f(V) \bowtie v) = \{H \mid H \in \mathbb{I}(\mathbb{D}^d) \text{ and } f(I(H, V)) \bowtie v\}.$$

Modal decision trees and modal random forests, when applied to d-dimensional datasets, may extract patterns written in $\mathcal{HS}^d$ [13, 14]. Therefore, establishing which variables and feature extraction functions, that is, which features, are informative in which hyperintervals, is a natural pre-processing step. Typical d-dimensional datasets are described by hundreds or thousands of variables in several dimensions, to which tens of feature extraction functions can be applied; this leads to a number of features in the order of the tens of thousands, from which a choice of a few units is usually made for further analysis.

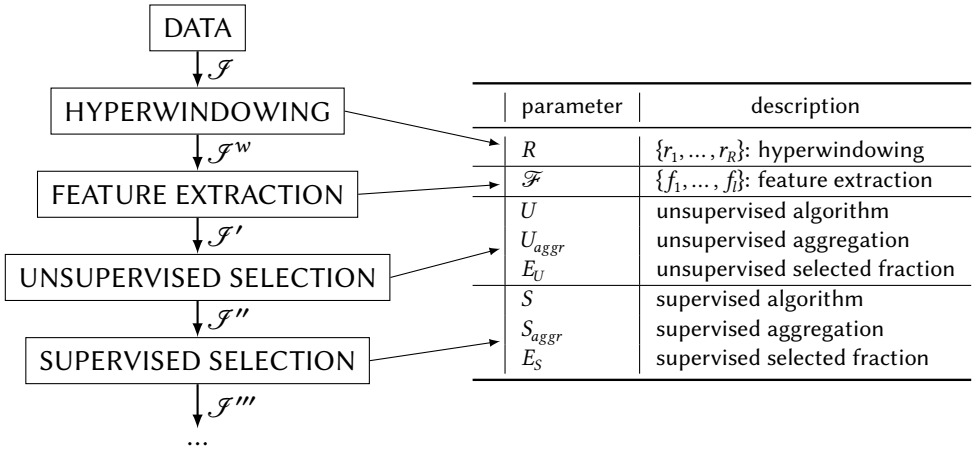


Figure 2: Schematic representation of the proposed pipeline and its parametrization.

3. Dimensional Feature Selection

Given a d-dimensional dataset in which every instance is described by N distinct points on each of the d dimensions leads to the need of considering $(N(N+1)/2)^d$ distinct hyperintervals; on each one of them, one should assess the value of n distinct variables under l distinct functions. A few considerations can be immediately drawn: *(i)* even in the context of filters, exploring all combinations seems infeasible, and *(ii)* hypothesis test-based selection should be avoided altogether, considering their probabilistic nature (that is, considering that performing too many probabilistic tests on the same data may lead to an unreliable result [17, 18]). However, the potential predictive power of features in d-dimensional data should be evaluated on hyperintervals, so that symbolic learning methods can take advantage of them.

As a tradeoff, let us introduce the notion of hyperwindow. Let $\mathbb{D}^d$ be a finite space defined as before, and let $H_1, \ldots, H_r$ be a set of hyperintervals such that, for every point $(x_1, \ldots, x_d) \in \mathbb{D}^d$, there is a hyperinterval H_i $(1 \leq i \leq r)$ with $(x_1, \ldots, x_d) \in H_i$; we call each such hyperinterval a *hyperwindow*, and the set $H_1, \ldots, H_r$ a *hyperwindowing* of $\mathbb{D}^d$. By extracting all features on all hyperwindows for a given r and evaluating them, one is able to approximate the behaviour of such features on a generic hyperinterval; there are, of course, several ways in which a hyperwindowing can be produced.

Based on the idea that a feature can be evaluated on a hyperwindow, unsupervised and supervised methods in the whole range of classic techniques can be systematically used to this end. Given a d-dimensional dataset $\mathcal{I}$, with n dimensional variables to which l feature extraction functions can be applied, and given a set of R hyperwindow parameterizations $\{r_1, \ldots, r_R\}$, a high-level description of the entire approach can be given in the following steps:

1. A d-dimensional dataset $\mathcal{I}^w$ is built by hyperwindowing the original one; $\mathcal{I}^w$ encompasses $\Sigma_{i=1}^{R} r_i l$ dimensional variables.
2. A tabular dataset is derived from $\mathcal{I}^w$, with $\Sigma_{i=1}^{R} r_i nl$ adimensional features. Each feature represents the result of applying a feature extraction function to a variable in a particular

hyperwindow, to which a *min-max* normalization is applied, to allow comparisons. This can be done in several ways: variable-wise, function-wise, function and window-wise, or across any combination of variables, depending on the nature of the dataset. Let $\mathcal{I}'$ be the result of this phase.

3. An univariate unsupervised filter U is applied to $\mathcal{I}'$. Since values are normalized, an *unsupervised individual score* (U_{score}) can be computed for each triple variable-function-hyperwindow. Selecting a pre-determined fraction of pairs based on their U_{score} produces a new tabular dataset $\mathcal{I}''$.
4. An univariate supervised filter S is then applied to $\mathcal{I}''$. As before, each feature-hyperwindow pair is assigned a *supervised individual score* (S_{score}). Selecting a pre-determined fraction of triples based on S_{score} produces a new tabular dataset $\mathcal{I}'''$.

It should be noted that the unsupervised and the supervised steps are actually independent of each other; however, the practice suggests that the unsupervised step should be applied first to *eliminate* the pairs that most probably do not contain any relevant information, and then, the supervised step (usually, computationally more expensive) should be applied to *select* the pairs that most probably contain information that is relevant to the problem. Furthermore, the above schema can be further extended by adding suitable *aggregation* functions: both unsupervised and supervised selection can be indeed performed after aggregating the triples either by feature extraction function, variable, hyperwindow, or a combination of them; in fact, it is convenient to assume that an aggregation function is always applied, whereas the simplest choice is an *identical* aggregation function with no effect. Other examples of aggregation functions include computing the maximum score or the average score across hyperwindows or across functions. Aggregation can be applied to the end result as well, in order to return the best variables, the best functions, and the best hyperwindows.

Summarizing, the following parameters should be set for a given set of experiments: the set of feature extraction functions ($\mathcal{F}$), the hyperwindowing parametrization ($\{r_1, \dots, r_R\}$), the unsupervised filter (U), the supervised filter (S), and two aggregation functions (U_{aggr}, S_{aggr}), along with the fraction of selected elements after the unsupervised step (E_U) and the fraction of selected elements after the supervised one (E_S). A schematic representation of our approach and its parametrization can be found in Fig. 2.

4. Experimental Evaluation

In order to evaluate the effectiveness of our approach we considered two cases, in the temporal ($d = 1$) and spatial ($d = 2$) setting, respectively. In both cases the datasets are public and had been used in the past for several other learning experiments.

In the temporal case, we use a dataset of audio samples, specifically of cough recordings, each one is labelled as *positive* or *negative* to COVID-19 infection [19]. The dataset is composed of 9986 audio samples recorded by 6613 volunteers, of which 235 declared to be positive. This data was originally used to derive smaller datasets, each posing a different form of the same task of COVID-19 diagnosis depending on the specific set of symptoms and conditions of the subjects. We refer to these tasks and datasets as TA1, TA2, and TA3. After the original

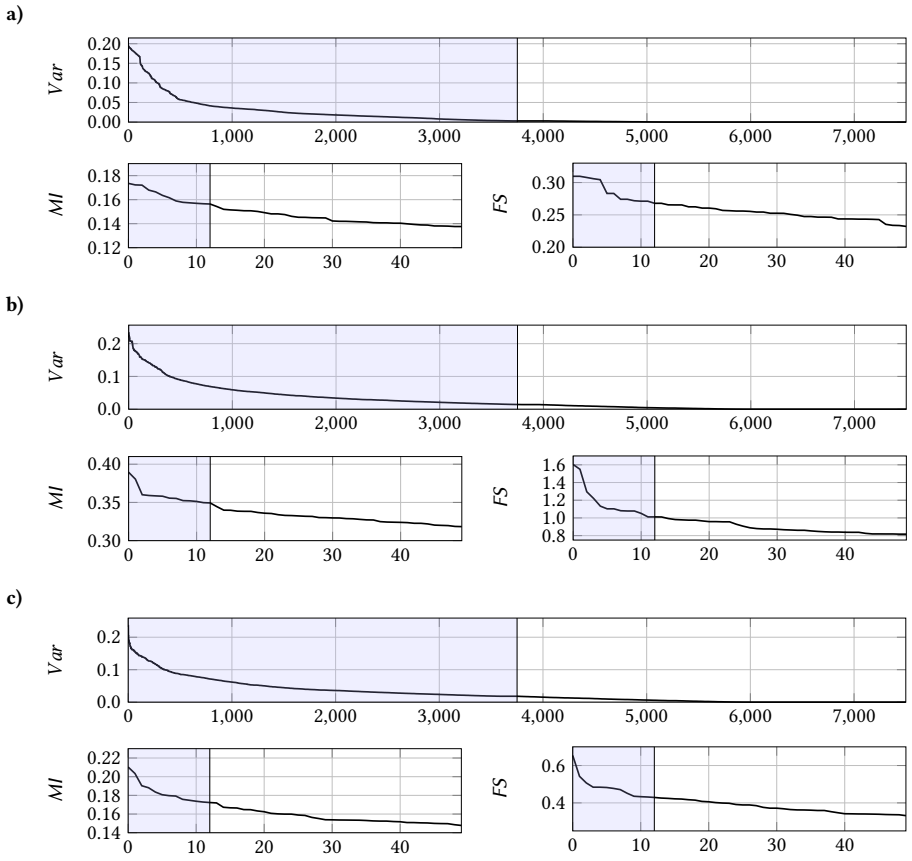


Figure 3: Summary graph of each feature selection step of the pipeline for each COVID-19 task: **a)** TA1, **b)** TA2, **c)** TA3. Each graph shows the score assigned to every feature sorted in non-ascending order. The blue area highlights the E_U (unsupervised) and E_S (supervised) selected features. The scores shown on the y axis are *Var* for Variance, *MI* for Mutual Information and *FS* for Fisher Score.

			1	2	3	4	5	6	7	8	9	10	11	12	13
TIME	TA1	*FS*	1.3e-17	1.3e-17	1.4e-16	1.0e-16	2.0e-16	3.6e-17	3.6e-17	6.6e-15	6.6e-15	2.3e-15	3.5e-16	3.5e-16	–
		MI	4.5e-16	2.2e-13	1.9e-10	5.6e-13	9.3e-15	3.4e-11	3.4e-11	3.4e-13	8.1e-17	1.3e-17	4.5e-15	6.6e-15	–
	TA2	*FS*	2.6e-9	4.1e-9	3.0e-9	8.2e-9	2.2e-9	2.3e-7	2.3e-7	1.3e-8	2.6e-7	2.6e-7	1.5e-8	5.8e-7	–
		MI	2.2e-9	1.5e-5	3.0e-9	1.4e-6	3.6e-6	2.1e-6	2.0e-7	5.9e-9	2.3e-7	5.6e-7	1.6e-6	5.0e-6	–
	TA3	*FS*	0.001	0.001	0.001	0.002	0.002	0.002	0.002	0.002	0.002	0.011	0.001	0.002	–
		MI	21.074	0.078	0.001	0.002	0.192	16.728	0.036	0.051	0.183	0.019	0.067	0.024	–
SPACE	IP	*FS*	9.0e-61	8.9e-61	8.9e-61	9.9e-61	9.0e-61	8.7e-61	1.0e-60	1.0e-60	1.0e-60	1.1e-60	1.1e-60	1.2e-60	–
		MI	1.7e-60	2.5e-59	1.5e-59	5.1e-60	6.6e-60	5.3e-60	4.9e-59	1.8e-60	4.4e-52	6.6e-60	9.3e-50	8.5e-51	–
	PC	*FS*	3.4e-56	1.8e-56	3.0e-56	2.2e-56	2.7e-56	3.7e-56	2.8e-56	2.8e-56	4.9e-56	3.0e-56	4.2e-56	5.3e-56	4.7e-56
		MI	3.7e-56	4.2e-56	2.7e-56	3.0e-56	1.2e-55	6.2e-56	3.0e-56	2.8e-56	1.2e-55	1.0e-55	4.8e-56	8.6e-56	3.4e-56
	PU	*FS*	8.3e-58	8.3e-58	8.3e-58	8.3e-58	8.3e-58	8.3e-58	8.3e-58	8.3e-58	8.3e-58	8.3e-58	8.4e-58	8.3e-58	8.4e-58
		MI	3.0e-56	2.3e-57	9.7e-58	1.4e-57	1.2e-57	1.6e-57	1.1e-56	2.9e-56	9.5e-58	1.4e-56	9.2e-58	1.5e-56	9.3e-58

Table 2
p-values of the selected features for COVID-19 (TIME) and LCC (SPACE) tasks.

publication, these tasks were approached and solved in several different way, among which

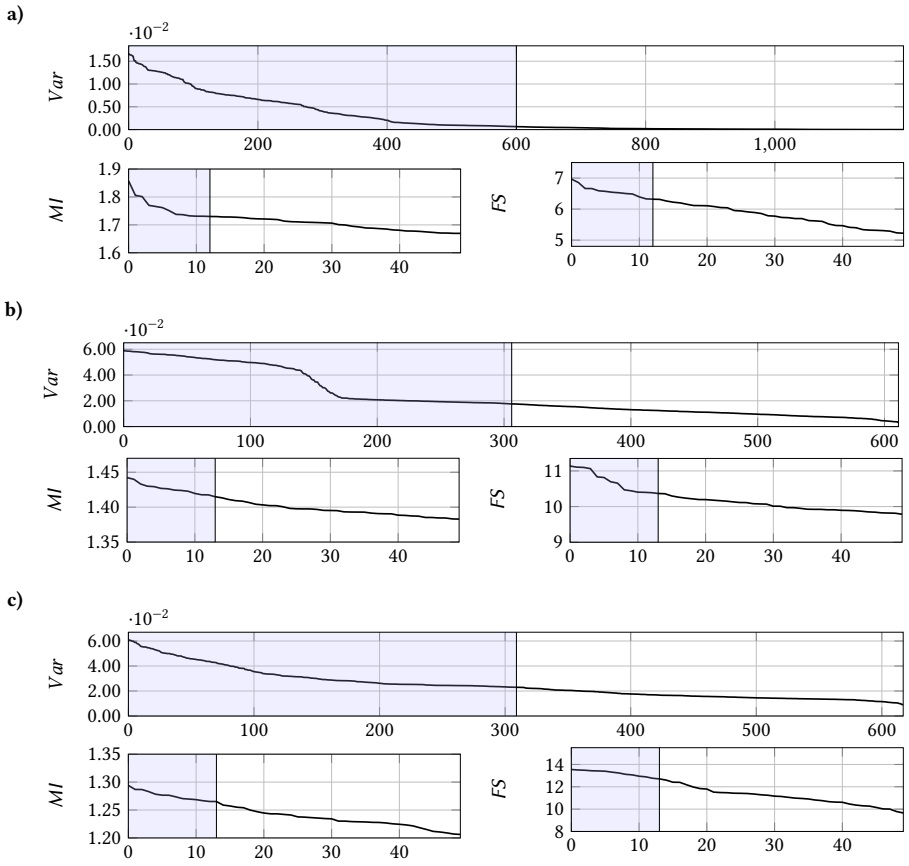


Figure 4: Summary graph of each feature selection step of the pipeline for each LCC task: **a)** *Indian Pines*, **b)** *Pavia Centre*, **c)** *Pavia University*. Each graph shows the score assigned to every feature sorted in non-ascending order. The blue area highlights the E_U (unsupervised) and E_S (supervised) selected features. The scores shown on the y axis are *Var* for Variance, *MI* for Mutual Information and *FS* for Fisher Score.

modal decision trees and forests [13]. In this paper, we considered the tasks TA1 ($m = 343$), TA2 ($m = 71$), and TA3 ($m = 55$). The temporal interpretation of audio signals emerges after the initial pre-processing via *Mel-Frequency Cepstral Coefficients* (*MFCC*) [20]. In short, MFCC consists of, first, separating the raw signal into chunks of small size and applying a *Discrete Fourier Transform (DFT)* to each of the chunks to produce a spectrogram of the sound at each chunk, second, converting and representing the frequency spectrum in *mel scale*, and third, convolving a set of n triangular band-pass filters across the frequency spectrum, discretizing it into a finite number of frequencies. Ultimately, this produces a multivariate time series representation where the n variables describe the power of the different sound frequencies; in this experiment, we fixed $n = 30$, that is, the variables are $F_1, \ldots, F_{30}$. For all these tasks we set $R = \{1, 3, 6\}$ for a total of 7 hyperwindows for each variable. There are several feature extraction functions that can be applied to temporal data. A commonly used portfolio is the so-called *Catch22* set [21], encompassing 22 selected functions that have shown the ability of

extracting relevant information from temporal data in several cases; to this set, we add *minimum, maximum*, and *mean*, leading to a set of $l = 25$ feature extraction functions.

As for the spatial case, we considered three multiclass datasets, that is, *Indian Pines* (IP), *Pavia University* (PU), *Pavia Centre* (PC), commonly used to benchmark methods for a problem known as *land cover classification* (LCC), which typically refers to the task of classifying pixels in remotely sensed images, associating each pixel with a label that captures the use of the land it depicts (e.g., *forest, urban area,* or *crop field*). Despite being an instance of standard image segmentation, this problem is usually dealt with as an image classification task. Images of this kind are usually captured from satellite or aerial sensors, and a single pixel carries the average electromagnetic response of a wide surface area; as such, the classification of a pixel in the image typically depends only on a close neighborhood of pixels around it. Additionally, the imagery is hyperspectral, that is, it encloses a large number of spatial variables (*spectral channels*), describing the strength of signals at different electromagnetic wavelengths. In the case of IP, we considered 1600 5×5 images ($m = 1600$) with $n = 200$ (in this cases, then, variables are named $C_1, \ldots, C_{200}$), each labelled with one of 16 classes ($k = 16$), while in the case of PC we had $n = 103$ and $k = 9$ ($m = 1600$, $C_1, \ldots, C_{103}$). In all cases we let $R = \{2\}$, and, in terms of feature extraction functions: in particular, we limited our search to *generalized mean*, *maximum*, and *minimum*, that is, $l = 3$.

For both the temporal and the spatial case we used a *variance*-based unsupervised filter (*Var*). This method is based on the idea that features with *too low variance* should be excluded. Examples of such filters include the *Variance Threshold method* in the `ScikitLearn` suite [22]. In this step we let U_{aggr} be the identity function (no aggregation was performed), selecting the first half of the total number of the variables ($E_U = 0.5$). In regard to the supervised selection we applied two different methods for both datasets: *Mutual Information* (*MI*) [23] and *Fisher Score* (*FS*) [24]. Depending on the problem and task we used different values for E_S: for all temporal dataset tasks we set $E_S = 0.032$ (so that the total number of selected features is 12); for the spatial datasets we used $E_S = 0.02$ for IP (the final selection resulting in 12 features) and $E_S = 0.04$ for both PC and PU (13).

All results are summarized in graphical form in Fig. 3 and 4. In both cases the scores in the unsupervised step decrease quite steeply well before reaching the first half of the set of analyzed features. This indicates that excluding as much as 50% of the features is quite effective, and that this fraction could be increased. After the second step, the value of the selected features can only be estimated: one way to do so is to use each feature in a pairwise, non-parametric hypothesis test (*validation* step). In the temporal case, the test is binary; in the spatial, multiclass case, instead, we performed one-versus-all tests to identify the class that is best distinguished from the other ones, and its corresponding p-value. It is crucial to observe that a hypothesis test could not be used in any preceding step as a selection criterium: given its probabilistic nature, performing too many tests renders its results unreliable [18]. This problem still exists during validation, though to a lesser extent. To compensate for it we applied the *Bonferroni* method for p-value correction; the corrected p-values are shown in Tab. 2. As it can be seen, most of the selected features, but not all, passed the test with confidence greater than 95%, and in most of those cases greater than 99%. The fact that not all the selected features are validated indicates that the choice is not trivial, and our approach was able to select a few very informative variable-function-hyperwindow triples from sets of between 600 and 7500. By way

of example, the first selection for TA1 in the case of *FS* corresponds to the *total power in lowest fifth of frequencies in the Fourier power spectrum* (from *Catch22*) applied to F_2 on the whole series, while for IP corresponds to the *mean value* of C_3 applied to the innermost 3×3 window.

5. Conclusions

Feature engineering and selection are a very well-known problem. Most, if not all the work, however, has been confined to the case of adimensional, tabular data. In this paper, we approached feature engineering and selection for dimensional data. Our proposal is inspired by recent results regarding symbolic learning from dimensional data, but it can be applied independently of any learning method as a way to study dimensional datasets. Conceived as a combination of known techniques, our framework is capable of identifying the most informative variables, the most informative feature extraction functions, and the most informative hyperwindows in a dimensional dataset of any number of dimensions. We performed a series of experiments in the case of two and three dimensions, obtaining quite encouraging results.

Our implementation, written in Julia, is open-source and it is part of a bigger project that includes symbolic learning techniques, feature selection, and post-hoc interpretation of symbolic models for non-tabular data.

Acknowledgments

We acknowledge the support of the project *Modal Geometric Symbolic Learning*, founded by the University of Ferrara under the FIRD program.

References

[1] H. M. Abdulwahab, S. Ajitha, M. A. N. Saif, Feature selection techniques in the context of big data: taxonomy and analysis, Applied Intelligence 52 (2022) 13568–13613.

[2] J. Cai, J. Luo, S. Wang, S. Yang, Feature selection in machine learning: A new perspective, Neurocomputing 300 (2018) 70–79.

[3] S. Solorio-Fernández, J. Carrasco-Ochoa, J. Martínez-Trinidad, A systematic evaluation of filter unsupervised feature selection methods, Expert Systems with Applications 162 (2020) 1–26.

[4] S. Huang, Supervised feature selection: A tutorial, Artificial Intelligence Research 4 (2015) 22–37.

[5] G. Chandrashekar, F. Sahin, A survey on feature selection methods, Computers & Electrical Engineering 40 (2014) 16–28.

[6] A. Gandomi, M. Haider, Beyond the hype: Big data concepts, methods, and analytics, International Journal of Information Management 35 (2015) 137–144.

[7] J. Rodríguez Diez, C. González, H. Boström, Boosting interval based literals, Intelligent Data Analysis 5 (2001) 245–262.

[8] Y. Yamada, E. Suzuki, H. Yokoi, K. Takabayashi, Decision-Tree Induction from Time-Series Data Based on a Standard-Example Split Test, in: Proceedings of the 12th International Conference on Machine Learning (ICML), 2003, p. 840–847.
[9] A. Brunello, E. Marzano, A. Montanari, G. Sciavicco, J48SS: A novel decision tree approach for the handling of sequential and time series data, Computers 8 (2019) 21.
[10] S. Jha, A. Tiwari, S. A. Seshia, T. Sahai, N. Shankar, TeLEx: learning signal temporal logic from positive examples using tightness metric, Formal Methods in System Design 54 (2019) 364–387.
[11] X. Li, C. Claramunt, A spatial entropy-based decision tree for classification of geographical information, Transactions in GIS 10 (2006) 451–467.
[12] D. Della Monica, G. Pagliarini, G. Sciavicco, I. Stan, Decision trees with a modal flavor, in: Proc. of the 221st Conference of the Italian Association for Artificial Intelligence (AIXA), volume 13796 of *LNCS*, Springer, 2022, pp. 47–59.
[13] F. Manzella, G. Pagliarini, G. Sciavicco, I. Stan, The voice of COVID-19: breath and cough recording classification with temporal decision trees and random forests, Artifcial Intelligence in Medicine 137 (2023) 1–14.
[14] G. Pagliarini, G. Sciavicco, Decision tree learning with spatial modal logics, in: Proc. of the 12th International Symposium on Games, Automata, Logics, and Formal Verification (GandALF), volume 346 of *EPTCS*, 2021, pp. 273–290.
[15] J. Halpern, Y. Shoham, A Propositional Modal Logic of Time Intervals, Journal of the ACM 38 (1991) 935–962.
[16] P. Balbiani, J. Condotta, L. Fariñas del Cerro, A model for reasoning about bidimensional temporal relations, in: Proc. of the 6th International Conference on Principles of Knowledge Representation and Reasoning (KR'98), Morgan Kaufmann, 1998, pp. 124–130.
[17] G. Heinze, D. Dunkler, Five myths about variable selection, Transplant International (2017) 6–10.
[18] M. Jafari, N. Ansari-Pour, Why, when and how to adjust your p values?, Cell 20 (2019) 604–607.
[19] C. Brown, J. Chauhan, A. Grammenos, J. Han, A. Hasthanasombat, D. Spathis, T. Xia, P. Cicuta, C. Mascolo, Exploring automatic diagnosis of COVID-19 from crowdsourced respiratory sound data, in: Proc. of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2020, pp. 3474–3484.
[20] S. Davis, P. Mermelstein, Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences, IEEE Transactions on Acoustics, Speech and Signal Processing 28 (1980) 357–366.
[21] C. Lubba, S. Sethi, P. Knaute, S. Schultz, B. Fulcher, N. Jones, Catch22: Canonical time-series characteristics - selected through highly comparative time-series analysis, Data Mining and Knowledge Discovery 33 (2019) 1821–1852.
[22] F. Pedregosa, *et. al.*, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.
[23] A. Kraskov, H. Stögbauer, P. Grassberger, Estimating mutual information, Physical review E 69 (2004) 066138.
[24] R. Duda, P. Hart, D. Stork, Pattern Classification, Wiley, 2012.