Attribute dependency data analysis for massive datasets by fuzzy transforms

Ferdinando Di Martino1 • Salvatore Sessa2

Accepted: 19 March 2021 / Published online: 15 April 2021 © The Author(s) 2021

Abstract
We present a numerical attribute dependency method for massive datasets based on the concepts of direct and inverse fuzzy transform. In a previous work, we used these concepts for numerical attribute dependency in data analysis: Therein, the multi-dimensional inverse fuzzy transform was useful for approximating a regression function. Here we give an extension of this method in massive datasets because the previous method could not be applied due to the high memory size. Our method is proved on a large dataset formed from 402,678 census sections of the Italian regions provided by the Italian National Statistical Institute (ISTAT) in 2011. The results of comparative tests with the well-known methods of regression, called support vector regression and multilayer perceptron, show that the proposed algorithm has comparable performance with those obtained using these two methods. Moreover, the number of parameters requested in our method is minor with respect to those of the cited in the above two algorithms.

Keywords
Attribute dependency • Data mining • Fuzzy transform

1 Introduction
Data analysis and data mining knowledge discovery processes represent powerful functionalities that can be combined in knowledge-based expert and intelligent systems in order to extract and build knowledge starting by data. In particular, attribute dependency data analysis is an activity necessary to reduce the dimensionality of the data and to detect hidden relations between features. Nowadays, in many application fields, data sources are massive (for example, web social data, sensor data, etc.), and it is necessary to implement knowledge extraction methods that can operate on massive data. Massive (Very Large (VL) and Large (L)) datasets (Chen and Zhang 2014) are produced and updated and they cannot be managed by traditional databases. Today, access via the Web to these datasets has led to develop technologies for managing them (cfr., e.g., (Dean 2014; Leskovec et al. 2014; Singh et al. 2015)).

We recall the regression analysis (cfr., e.g., (Draper and Smith 1988; Han et al. 2012; Johnson and Wichern 1992; Jun et al. 2015; Piatecky–Shapiro and Frawley 1991)) for estimating relationships among variables in the datasets (cfr., e.g., (Lee and Yen 2004; Mitra et al. 2002; Tanaka 1987; Wood et al. 2015)) and fuzzy tools for attribute dependency (Vucetic et al. 2013; Yen and Lee 2011).

Machine learning soft computing models were proposed in the literature to perform nonlinear regressions on high dimensional data; two well-known machine learning nonlinear regression algorithms are support vector regression (SVR) (Drucker et al. 1996) and multilayer perceptron (MLP) (cfr., e.g., (Collobert and Bengio 2004; Cybenko 1989; Hastie et al. 2009; Haykin 1999,2009; Murtagh 1991; Schmidhuber 2014)) algorithms. The main problems of these algorithms are the complexity of the model due to the presence of many parameters to be set by the user, and the presence of overfitting, phenomenon in which the regression function fits optimally the training set data, but
fails in predictions on new data. K-fold cross-validation
techniques are proposed in the literature to avoid overfitting (Anguita et al. 2005). In Thomas and Suhner (2015), a
pruning method based on variance sensitivity analysis is
proposed to find the optimal structure of a multilayer perceptron in order to mitigate overfitting problems. In Han
and Jian (2019), a novel sparse-coding kernel algorithm is
proposed to overcome overfitting in disease diagnosis.

Some authors proposed variations of nonlinear machine
learning regression models to manage massive data. In
Cheng et al. (2010), Segata and Blanzieri (2009) a fast-
local support vector machine (SVM) method to manage large datasets are presented in which a set of multiple local
SVMs for low-dimensional data are constructed. In Zheng
et al. (2013), the authors proposed an incremental version of
the vector machine regression model to manage large-
scale data. In Peng et al. (2013) the authors proposed a
parallel architecture of a logistic regression model for
massive data management. Recently, variations of the
extreme learning machine (ELM) regression methods for
massive data based on the MapReduce model are presented
(Chen et al. 2017; Yao and Ge 2019).

The presence of a high number of parameters makes
SVR and MLP methods too complex to be integrated as
components into an intelligent or expert system. In this
research, we propose a model of attribute dependency in
massive datasets based on the use of the multi-dimensional
fuzzy transform. We extend the attribute dependency
method presented in Martino et al. (2010a) to massive
data sets in which the inverse multi-dimensional fuzzy
transform is used as a regression function. Our goal is to
guarantee a high performance of the proposed method in
the analysis of massive data, maintaining, at the same time,
the usability of the previous multi-dimensional fuzzy
transform attribute dependency. As proposed by Jun et al. (2015), we
use a random sampling algorithm for subdividing the
dataset in subsets of equal cardinality.

The fuzzy transform (F-transform) method (Perfilieva
2006) is a technique which approximates a given function
by means of another function unless an arbitrary constant.
This approach is particularly flexible in the applications
such as image processing (cfr., e.g., (Martino et al.
2008,2010b,2011b; Martino and Sessa 2007,2012)), data
analysis (cfr., e.g., (Martino et al. 2010a,2011a;Perfilieva
et al. 2008)). In this last work, an algorithm, called FAD
(F-transform Attribute Dependence), evaluates an attribute
Xz depending from k attributes X1,...Xk (predictors) with
z $\in$ {1,...,k}, i.e. $X_z = H(X_1,...X_k)$, and the (unknown)
function H is approximated with the inverse multi-dimen-
sional F-transform via a procedure presented in Perfilieva
et al. (2008). The error of this approximation in Martino
et al. (2010a) is measured from a statistical index of
determinancy (Draper and Smith 1988; Johnson and
Wichern 1992). If it overcomes a prefixed threshold, then
the functional dependency is found. Each attribute has an
interval $X_i = [a_i,b_i]$, i = 1,..., k, as domain of knowledge.
Then an uniform fuzzy partition (whose definition is given
in Sect. 2) of fuzzy sets {$A_1,A_2,...,A_m$} defined on $[a_i,b_i]$
is created assuming $n_i \geq 3$.

The main problem in the use of the inverse F-transform
for approximating the function $H$ consists in the fact that
the data are not sufficiently dense with respect to the fuzzy
partitions. The FAD algorithm solves this problem with an
iterative process which is shown in Sect. 3. If the data are
not sufficiently dense with respect to the fuzzy partitions,
the process stops otherwise an index of determinancy is
calculated. If this index is greater than a threshold $\alpha$,
the functional dependency is found and the inverse F-transform
is considered as approximation of the function $H$, otherwise
a finer fuzzy partition is set with $n_i = n_i + 1$. The FAD
algorithm is schematized in Fig. 1.

In this paper, we propose an extension of the FAD
algorithm, called MFAD (massive F-transform attribute
dependency) for finding dependencies between numerical
attributes in massive datasets. In other words, by using a
uniform sampling method, we can apply the algorithm of
Martino et al. (2010a) to several sample subsets of the data
and hence we extend the results obtained to the overall
dataset with suitable mathematical artifices.

Indeed, the dataset is partitioned randomly in $s$ subsets
having equal cardinality to which we apply the F-transform
method.

Let $D_i = [a_{i1},b_{i1}] \times \cdots \times [a_{il},b_{il}]$, $i = 1,...,s$, be the
Cartesian product of the domains of the attributes $X_1,$
$X_2,$...,$X_k$, where $a_d$ and $b_d$ are the minimum and maximum
values of $X_i$ in the $i$th subset. Hence, the multi-dimensional
inverse F-transform $H^F_{n_1,n_2,...,n_l}$ is calculated for approxi-
mating the function $H$ in the domain $D_i$ and an index of
determinancy $r^2_{cl}$ is calculated for evaluating the error in the
approximation of $H$ with $H^F_{n_1,n_2,...,n_l}$ in $D_i$. For simplicity, we
put $n_{ij} = n_{i2} = \cdots = n_{ik} = n_1$ and thus $H^F_{n_1,n_2,...,n_l} = H^F_{n_1}$. In
order to obtain the final approximation of $H$, we introduce
weights for considering the contribute of the inverse F-
transform $H^F_{n_1}$ in the approximation of $H$. We calculate the
weighted mean of $H^F_{n_1},...H^F_{n_1}$ replacing the weights with
the indices of determinancy $r^2_{c1},...,r^2_{cs}$.

Calculate the approximated value of $H^F$ in $(x_1,...,x_k) \in
\bigcup_{i=1}^s D_i$ given by

$$H^F(x_1,x_2,...,x_k) = \frac{\sum_{i=1}^s w_i(x_1,x_2,...,x_k) \cdot H^F_{n_1}(x_1,x_2,...,x_k)}{\sum_{i=1}^s w_i(x_1,x_2,...,x_k)}$$

(1)

where

$\square$ Springer
For example, we consider two attributes, $X_1$ and $X_2$, as inputs and suppose, for simplicity, that the dataset is partitioned in two subsets. Figure 2 shows two rectangles $D_1$ (red) and $D_2$ (green). The zone labeled as $A$ of the input space is covered by the domain $D_2$: in this zone the weight $w_1$ is null and $F = F_2$. Conversely, in the zone $C$ the contribution of $F_2$ is null and $F = F_1$. In the zone labeled as $B$, the inverse F-transforms, calculated for both subsets, contribute to the final evaluation of $H$, with a weight corresponding to the index of determinacy.

Figure 3 shows the schema of MFAD. We apply our method on a $L$ dataset loadable in memory, so we can apply also the method of Martino et al. (2010a) and hence we compare the results obtained by using both methods. As test dataset, we consider the last Italian census data acquired during 2011 by ISTAT (Italian National Statistical Institute). Section 2 contains the F-transform in one and more variables (Perfilieva et al. 2008). In Sect. 3, the F-transform attribute dependency method is presented, Sect. 4 contains the results of our tests. Conclusions are described in Sect. 5.

### 2 F-transforms in one and $k$ variables

Following the definitions of Perfilieva (2006). We recall the main notations for making this paper self-contained. Let $n \geq 2$, $x_1$, $x_2$, ..., $x_n$ be points (nodes) of $[a,b]$. $x_1 = a < x_2 < \cdots < x_n = b$. The fuzzy sets $A_1$, ..., $A_n$: $[a,b] \rightarrow [0,1]$ (basic functions) constitute a fuzzy partition of $[a,b]$ if $A_i(x_i) = 1$ for $i = 1,\ldots, n$; $A_i(x) = 0$ if $x \not\in [x_{i-1}, x_i]$ for $i = 2,\ldots, n$; $A_i(x)$ is a continuous on $[a,b]$; $A_i(x)$ strictly increases on $[x_{i-1}, x_i]$ for $i = 2,\ldots, n$ and strictly decreases on $[x_{n-1}, x_n]$ for $i = 1,\ldots, n-1$; $\sum_{i=1}^{n} A_i(x) = 1$ for every $x \in [a,b]$. The partition $\{A_1(x), \ldots, A_n(x)\}$ is said uniform if $n \geq 3$, $x_1 = a + h(i-1)$, where $h = (b-a)/(n-1)$ and $i = 1, 2,\ldots, n$ (equidistance); $A_i(x) = A_i(x_i + x)$ for $x$...
Perfilieva et al. (2008). The F-transforms can be extended to \( f_{\text{F}} \) for every \( f \in \mathcal{F} \) with respect to \( \{A_1, A_2, \ldots, A_n\} \), that is for every \( i \in \{1, \ldots, n\} \) there exists an index \( j \in \{1, \ldots, m\} \) such that \( A_i(p_j) > 0 \), then the \( n \)-tuple \( [F_1, F_2, \ldots, F_n] \) is the discrete direct F-transform of \( f \) with respect to \( \{A_1, A_2, \ldots, A_n\} \), where each \( F_i \) is given by

\[
F_i = \frac{\sum_{j=1}^{m} f(p_j)A_i(p_j)}{\sum_{j=1}^{m} A_i(p_j)}
\]

for \( i = 1, \ldots, n \). Then we define the discrete inverse F-transform of \( f \) with respect to the basic functions \( \{A_1, A_2, \ldots, A_n\} \) by setting

\[
f_{\text{F}, n}(p_j) = \sum_{i=1}^{n} F_i A_i(p_j)
\]

for every \( j \in \{1, \ldots, m\} \). Now we recall concepts fromPerfilieva et al. (2008). The F-transforms can be extended to \( k \geq 2 \) variables considering the Cartesian product of intervals \( [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_k, b_k] \). Let \( x_{11}, x_{12}, \ldots, x_{1n} \in [a_1, b_1], \ldots, x_{k1}, x_{k2}, \ldots, x_{kn} \in [a_k, b_k] \) be \( n_1 + \cdots + n_k \) assigned points (nodes) such that \( x_{ij} = a_i \) \( < x_{i2} < \cdots < x_{in_i} = b_i \) and \( \{A_{i1}, A_{i2}, \ldots, A_{in_i}\} \) be a fuzzy partition of \( [a_i, b_i] \) for \( i = 1, \ldots, k \). Let the function \( f(x_1, x_2, \ldots, x_k) \) be assuming values in \( m \) points \( p_j = (p_{j1}, p_{j2}, \ldots, p_{jk}) \in [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_k, b_k] \) for \( j = 1, \ldots, m \). The set \( P = \{p_{j1}, p_{j2}, \ldots, p_{jk}\} \) is said sufficiently dense with respect to \( \{A_{i1}, A_{i2}, \ldots, A_{in_i}\} \). When \( k = 2 \), the \( (h_1, h_2) \)th component \( F_{h_1h_2} \) of the discrete direct F-transform of \( f \) with respect to \( \{A_{i1}, A_{i2}, \ldots, A_{in_i}\} \) is defined as

\[
F_{h_1h_2} = \frac{\sum_{j=1}^{m} f(p_{j1}, p_{j2}) \cdot A_{h_1}(p_{j1}) \cdot A_{h_2}(p_{j2})}{\sum_{j=1}^{m} A_{h_1}(p_{j1}) \cdot A_{h_2}(p_{j2})}
\]
Thus we define the discrete inverse F-transform of $f$ with respect to $\{A_{11}, A_{12}, \ldots, A_{1m_1}\}, \ldots, \{A_{k1}, A_{k2}, \ldots, A_{km_k}\}$ by setting for $p_j = (p_{j1}, p_{j2}, \ldots, p_{jk}) \in [a_1, b_1] \times \ldots \times [a_k, b_k]$:

$$f^c_{n_{i1}, \ldots, n_{i2}}(p_{j1}, p_{j2}, \ldots, p_{jk}) = \sum_{h_{11}=1}^{n_{i1}} \ldots \sum_{h_{k1}=1}^{n_{ik}} F_{h_{1k}, \ldots, h_{k1}} \cdot A_{h_{1k}}(p_{j1}) \cdot \ldots \cdot A_{h_{k1}}(p_{jk})$$  \hspace{2cm} (6)

for $j = 1, \ldots, m$. The following Theorem holds (Perfilieva 2006):

**Theorem 1** Let $f(x_1, x_2, \ldots, x_k)$ be a function assigned on the set of points $P = \{p_{j1}, p_{j2}, \ldots, p_{jk}\} \in P$, $j = 1, \ldots, m$, the following inequality holds:

$$\left| f(p_{j1}, p_{j2}, \ldots, p_{jk}) - f^c_{n_{i1}(c), n_{i2}(c), \ldots, n_{ik}(c)}(p_{j1}, p_{j2}, \ldots, p_{jk}) \right| < \varepsilon$$  \hspace{2cm} (8)

### 3. Multi-dimensional algorithm for massive datasets

#### 3.1 FAD algorithm

We schematize a dataset in tabular form as

| $X_1$ | $\ldots$ | $X_i$ | $\ldots$ | $X_r$ |
|-------|--------|--------|--------|--------|
| $O_1$ | $p_{i1}$ | $\ldots$ | $p_{ij}$ | $p_{ir}$ |
| $\ldots$ | $\ldots$ | $\ldots$ | $\ldots$ | $\ldots$ |
| $O_m$ | $p_{ml}$ | $\ldots$ | $p_{mi}$ | $p_{mr}$ |

Here $X_1, \ldots, X_i, \ldots, X_r$ are the involved attributes and $O_1, \ldots, O_r, O_m$ $(m > r)$ are the instances and $p_{ij}$ is the value of the attribute $X_i$ for the instance $O_j$. Each attribute $X_i$ can be considered as a numerical variable assuming values in the domain $[a_i, b_i]$, where $a_i = \min\{p_{1i}, \ldots, p_{mi}\}$ and $b_i = \max\{p_{1i}, \ldots, p_{mi}\}$. We analyze the functional dependency between attributes in the form:

$$X_z = H(X_{1z}, \ldots, X_{kz})$$  \hspace{2cm} (9)

where $z \in \{1, \ldots, r\}$, $k \leq r \leq m$, $X_z \neq X_1, X_2, \ldots, X_k$, $H: [a_1, b_1] \times [a_2, b_2] \times \ldots \times [a_k, b_k] \rightarrow [a_z, b_z]$ is continuous. Let $X_{(j-1)}, X_{(j+1)}$ be a function assigned on the set $P$.

### 3.2 Update algorithm

By setting $H = \{p_{j1}, p_{j2}, \ldots, p_{jk}\} = p_{jc}$ for $j = 1, 2, \ldots, m$, the components of $H$ are given by

$$H_{h_{1k}, \ldots, h_{k1}} = \frac{\sum_{j=1}^{m} p_{jc} \cdot A_{h_{1k}}(p_{j1}) \cdot \ldots \cdot A_{h_{k1}}(p_{jk})}{\sum_{j=1}^{m} A_{h_{1k}}(p_{j1}) \cdot \ldots \cdot A_{h_{k1}}(p_{jk})}$$  \hspace{2cm} (11)
The inverse F-transform $H_{n_{j_1...j_k}}^F$ is defined as

$$H_{n_{j_1...j_k}}^F(p_{j_1}, p_{j_2}, ..., p_{j_k}) = \sum_{h_1=1}^{n_1} \sum_{h_2=1}^{n_2} \cdots \sum_{h_k=1}^{n_k} F_{h_1...h_k} \cdot A_{1h_1} (p_{j_1}) \cdot \cdots \cdot A_{kh_k} (p_{j_k})$$

(12)

The error of the approximation is evaluated in $(p_{j_1}, ..., p_{j_m})$ by using the following statistical index of determinacy (Draper and Smith 1988; Johnson and Wichern 1992):

$$r_c^2 = 1 - \left(1 - r_c^2\right) \cdot \frac{m-1}{m-k-1}$$

(14)

The pseudocode of the algorithm FAD is schematized below.

---

**Function FAD**

**Input:**
- $X_z = H(X_1, ..., X_k)$
- DT - Dataset composed by m instances with attributes $X_1, X_2, ..., X_r$
- $\alpha$ - threshold value

**Output:**
- Direct F-transform components $F_{h_1...h_k}$
- Index of determinacy $r_c^2$  

1. n := 3  
2. $r_c^2 = 0$  
3. **DO WHILE** $r_c^2 \leq \alpha$  
4. Set F as a matrix of dimension $n^k$  
5. F = {0} //initialize to 0 the components of F  
6. FOR each combination $\{h_1, ..., h_k\}$  
7. $F[h_1,...,h_k] = \text{DirectFuzzyTransformComponent(DT, n, k,z,h[1,...,k] )} // calculate the F-Transform component $F_{h_1...h_k}$  
8. NEXT $\{h_1, ..., h_k\}$  
9. $r_c^2 = \text{IndexofDeterminacy(DT, n, k,z, F)} // Calculate the index of determinacy $r_c^2$  
10. n := n+1  
11. **END DO**  
12. **RETURN** F, $r_c^2$  
13. **END IF**  
14. **END**

$$r_c^2 = \frac{\sum_{j=1}^{m} \left(H_{n_{j_1...j_k}}^F(p_{j_1}, p_{j_2}, ..., p_{j_k}) - \hat{p}_z\right)^2}{\sum_{j=1}^{m} (p_{j_k} - \hat{p}_z)^2}$$

(13)

where $\hat{p}_z$ is the mean of the values of the attribute $X_z$. If $r_c^2 = 0$ (resp., $r_c^2 = 1$) means that (11) does not fit (resp., fits perfectly) to the data. However we use a variation of (11) for taking into account both the number of independent variables and the scale of the sample used (Martino et al. 2010a) given by

$$r_c^2 = 1 - \left(1 - r_c^2\right) \cdot \frac{m-1}{m-k-1}$$

The function DirectFuzzyTransform() is used to calculate each direct F-transform component. The function BasicFunction() calculates the value $A_{ih_i}(x)$ for an assigned $x$ of the $h_i$th basic function of the $i$th fuzzy partition. IndexofDeterminacy calculates the index of determinacy.
**Function DirectFTransformComponent**

**Description**
The component of the direct F-transform $F_{h_2...h_k}$

**Input:**
- DT - dataset analysed
- n - number of fuzzy sets of the fuzzy partitions
- k - number of the input attributes
- z - index of the output attribute $X_z$
- $h[1,...,k]$ = array of indices of the combination $\{h_1,..., h_k\}$

**Output:**
- Direct F-transform component $F_{h_2...h_k}$
- 0 if the data points are not sufficiently dense with respect to the fuzzy partition

```
1  Num = 0, Denum = 0, j
2  FOR each p in DT
3     val = 1
4     xz = p.X[z] // value of the attribute $X_z$
5     FOR i = 1 to k
6         a = min(DT.X[i]) // inf of [a_i,b_i]
7         b = max(DT.X[i]) // sup of [a_i,b_i]
8         j = h[i] // index of the $h^{th}$ fuzzy set of the fuzzy partition of [a,b]
9         x = p.X[i] // value of the $i^{th}$ attribute $X_i$
10        A = BasicFunction(a,b,n,x,j)
11       val = val*A
12     NEXT i
13    H = val*F
14  Denom = Denom + val
15  NEXT p
16  IF Denom = 0 RETURN 0
17  ELSE RETURN Num/Denum
18  END IF
19  END
```
**Function BasicFunction**

| Description | An uniform fuzzy partition is created for the interval [a,b] |
|-------------|----------------------------------------------------------|
| **Input:**  | a - inf value of the interval [a,b]                     |
|             | b - sup value of the interval [a,b]                     |
|             | n - number of fuzzy sets in the fuzzy partition         |
|             | x - value of x                                          |
|             | k - index of the kth fuzzy set of the fuzzy partition   |
| **Output:** | Return the basic function value $A_k(x)$                |

1. Set $h = (b - a)/(n - 1)$
2. Set $x[1..n]$
3. $x[1] = a$
4. FOR $i = 2$ to $n$
5. $x[i] = x[i-1] + h$
6. NEXT $i$
7. IF $k = 1$ THEN $x[k-1] = x[1]$
8. $x_{min} = x[1]$
9. $x_{max} = x[k+1]$
10. ELSE IF $k = n$ THEN $x[k-1] = x[1]$
11. $x_{min} = x[k-1]$
12. $x_{max} = x[k]$
13. ELSE
14. $x_{min} = x[k-1]$
15. $x_{max} = x[k+1]$
16. ENDIF
17. ENDIF
18. Set $A = 0$
19. IF $x_{min} \leq x \leq x_{max}$ THEN
20. \[ A = 0.5 \cdot (1 + \cos \frac{\pi}{h} (x - x[i])) \]
21. END IF
22. RETURN $A$
23. END
### 3.2 MFAD algorithm

We consider a massive dataset DT composed by $r$ attributes where $X_1, \ldots, X_r$, and $m$ instances $O_1, \ldots, O_m$ ($m > r$). We make a partition of DT in $s$ subsets DT$_1, \ldots, $ DT$_s$, with the same cardinality, by using an uniform random sample in such a way each subset is loadable in memory. We apply the FAD algorithm to each subset, calculating the direct F-transform components, the inverse F-transforms $H^F_{n1} \ldots H^F_{ns}$, the indices of determinacy $r^2_{c1}, \ldots,$ $r^2_{cs}, r^2_{c}$ and the domains $D_1, \ldots, D_s$, where $D_l = [a_{l1}, b_{l1}] \times \ldots \times [a_{l2}, b_{l2}], \ l = 1, \ldots, s$. All these quantities are saved in memory. If a dependency $f$ is not found for the $l$th subset, the corresponding value of $r^2_{cl}$ is set to 0. The pseudocode of MFAD is given below.

| Function IndexofDeterminacy |
|-------------------------------|
| **Description** | The index of determinacy $r^2_{c}$ is calculated |
| **Input:** | DT - dataset analyzed |
| | n - number of fuzzy sets of the fuzzy partitions |
| | k - number of the input attributes |
| | z - index of the of the output attribute $X_2$ |
| | F - matrix of the direct F-transform components |
| **Output:** | Direct F-transform component $F_{h_1h_2\ldots h_k}$ |
| | 0 if the data points are not sufficiently dense with respect to the fuzzy partition |

```
1 Num = 0, Denum = 0, m = 0
2 mz= mean(X[z]) // mean value of the attribute $X_2$ in DT
3 FOR each p in DT
4   m = m + 1 //in m calculated the number of instances in DT
5   val = 1
6   xz = p.X[z] // value of the attribute $X_2$
7   FOR each combination $\{h_1..h_k\}$
8     val = 1
9     FOR i = 1 to k
10        b = max(DT.X[i]) // sup of $[a_i,b_i]$
11        j = h[i] // index of the h-th fuzzy set of the fuzzy partition of $[a,b]$
12        x = p.X[i] // value of the i-th attribute $X_i$
13        A = BasicFunction(a,b,n,x[j])
14        val = val*A
15     NEXT i
16     H = H + val* F[h_1..h_k]
17     NEXT $\{h_1..h_k\}$
18     num = num + (H - mz)^2
19     denum = denum + (xz - mz)^2
20    NEXT p
21 RETURN (Num/Denum)*(m-1)/(m-k-1)
22 END
```
Now we consider a point \((x_1, x_2, \ldots, x_k)\) \(\in S\). In order to approximate the function \(H(x_1, x_2, \ldots, x_k)\), we calculate the weights as:

\[
w_i(x_1, x_2, \ldots, x_k) = \begin{cases} \frac{r_i}{\alpha_i} & \text{if } (x_1, x_2, \ldots, x_k) \in D_i, \quad i = 1, \ldots, s \\ 0 & \text{otherwise} \end{cases}
\]

(15)

If for any subset the functional dependency is not found, then \(w_i = 0\) for each \(i = 1, \ldots, s\). Otherwise, the approximated value of \(H(x_1, x_2, \ldots, x_k)\) is given by

\[
H^F(x_1, x_2, \ldots, x_k) = \frac{\sum_{i=1}^{s} w_i(x_1, x_2, \ldots, x_k) \cdot H_i^F(x_1, x_2, \ldots, x_k)}{\sum_{i=1}^{s} w_i(x_1, x_2, \ldots, x_k)}
\]

(16)

which is also the value of \(X_z\). To analyze the performance of the MFAD algorithm, we execute a set of experiments on a large dataset formed from 402,678 census tracts of the Italian regions provided by the Italian National Statistical Institute (ISTAT) in 2011. Therein, 140 numerical attributes belong to each of the following categories:

- Inhabitants,
- Foreigner and stateless inhabitants,
- Families,
- Buildings,
- Dwellings.

The FAD method is applied on the overall dataset, the MFAD method is applied by partitioning the dataset in \(s\) subsets, and we perform the tests varying the value of the parameter \(s\) and by setting the threshold \(\alpha = 0.7\).

In addition, we compare the MFAD algorithm with the support vector regression (SVR) and multilayer perceptron (MLP) algorithms.

### 4 Experiments

Table 1 shows the 402,678 census tracts of Italy divided for each region. Table 2 shows the approximate number of census tracts in each subset for each partition of the dataset in \(s\) subsets.

In any experiment, we apply the MFAD algorithm to analyze the attribute dependency explored of an output attribute \(X_z\) from a set of input attributes \(X_1, X_2, \ldots, X_r\). In all the experiments, we set \(\alpha = 0.7\) and partition randomly the dataset in \(s\) subsets. We now show the results obtained in three experiments.

#### 4.1 Experiment A

In this experiment, we explore the relation between the density of resident population with laurea degree and the density of resident population employed. Generally speaking, a higher density of population with laurea degree should correspond to a greater density of population employed. The attribute dependency explored is \(H_z = H(X_1)\), where

- Input attribute: \(X_1 =\) Resident population with laurea degree
- Output attribute: \(X_z =\) Resident population over 15 employed
We apply the FAD algorithm on different random subsets of the dataset, and then we calculate the index of determinacy (12). In Table 3, we show the value of the index of determinacy \( r_{cl}^2 \) obtained for different values of \( s \). For \( s = 1 \), we have the overall dataset.

The results in Table 3 show that the dependency has been found. We obtain \( r_{cl}^2 = 0.760 \) by using FAD algorithm on the entire dataset, while the best value of \( r_{cl}^2 \) (reached by using MFAD) is 0.758 for \( s = 16 \). Hence the related smallest difference between the two algorithms is 0.02. Figure 4 shows in abscissas the input \( X_1 \) and in ordinates the output \( H^F(x_1) \) for \( s = 1, 10, 16, 40 \).

### 4.2 Experiment B

In this experiment, we explore the relation between the density of residents with job or capital income and the density of families in owned residences. We expect that the greater the density of residents with job or capital income is, the resident families density in owned homes the greater is. The attribute dependency explored is \( H_z = H(X_1) \), where:

- Input attributes: \( X_1 \) = Resident population with job or capital income
- Output attribute \( X_z \) = Families in owned residences
- After some tests, we put \( \alpha = 0.8 \).

Table 4 shows \( r_{cl}^2 \) obtained for different values of \( s \):
- \( r_{cl}^2 = 0.881 \) in FAD algorithm on the entire dataset,
- \( r_{cl}^2 = 0.878 \) in MFAD obtained for \( s = 13, 16 \). The smallest index of dependency difference is 0.003.

Figure 5 shows in abscissas the input \( X_1 \) and in ordinates the output \( H^F(x_1) \) for \( s = 1, 10, 16, 40 \).

### 4.3 Experiment C

In this experiment, the attribute dependency explored is \( H_z = H(X_1, X_2) \), where

- Input attributes:
  - \( X_1 \) = Density of residential buildings built with reinforced concrete
  - \( X_2 \) = Density of residential buildings built after 2005
- Output attribute:
• $X_c$ = Density of residential buildings with state of good conservation

After some tests, we decided $z = 0.75$ in this experiment. In Table 5, we show $r_{cl}^2$ obtained for different values of $s$. $r_{cl}^2 = 0.785$ in FAD algorithm on the entire dataset. $r_{cl}^2 = 0.781$ in MFAD algorithm obtained for $s = 13, 16$. The smallest index of dependency difference is 0.004.

Now we present the results obtained by considering all the experiments performed on the entire dataset in which the dependency was found ($r_{cl}^2 > 0.7$). We consider the index of determinacy in the FAD algorithm ($s = 1$) and the minimum and maximum values of the index of determinacy obtained by using the MFAD algorithm for $s = 9, 10, 11, 13, 16, 20, 26, 40$.

A functional dependency was found in 43 experiments. Figure 6 (resp., 7) shows the trend of the difference between the maximum (resp., minimum) value calculated for $r_{cl}^2$ in MFAD and in FAD for the same experiment. In abscissae, we have $r_{cl}^2$ in the FAD method, in ordinates the difference between the two indices. For all the experiments this difference is always below 0.005 (resp., 0.0015).

These results show that the MFAD algorithm is comparable with the FAD algorithm, independently of the choice of the number of subsets partitioning the entire dataset (Fig. 7).

Figure 8 shows the mean CPU time gain obtained by MFAD algorithm with different partitions, with respect to the CPU time obtained by using FAD algorithm ($s = 1$). The CPU time gain is given by the difference between the CPU time measured by using $s = 1$, and the CPU time measured by using a partition in $s$ subsets, divided by the CPU time measured for $s = 1$. The CPU time gain is

---

### Table 1: Number of census tracts for each Italian region

| ID region | Description | Number of census tracts |
|-----------|-------------|-------------------------|
| 001       | Piemonte    | 35,672                  |
| 002       | Valle d’Aosta | 1902                |
| 003       | Lombardia   | 53,173                  |
| 004       | Trentino Alto Adige | 11,712 |
| 005       | Veneto      | 33,883                  |
| 006       | Friuli Venezia Giulia | 8278    |
| 007       | Liguria     | 11,054                  |
| 008       | Emilia Romagna | 38,603          |
| 009       | Toscana     | 28,917                  |
| 010       | Umbria      | 7480                    |
| 011       | Marche      | 11,862                  |
| 012       | Lazio       | 32,065                  |
| 013       | Abruzzo     | 9529                    |
| 014       | Molise      | 2821                    |
| 015       | Campania    | 24,323                  |
| 016       | Puglia      | 22,514                  |
| 017       | Basilicata  | 5107                    |
| 018       | Calabria    | 13,121                  |
| 019       | Sicilia     | 36,681                  |
| 020       | Sardegna    | 13,981                  |

### Table 2: Number of census tracts for each subset by varying $s$

| $s$ | Number of census tracts |
|-----|-------------------------|
| 8   | $5.0\times10^4$         |
| 9   | $4.5\times10^4$         |
| 10  | $4.0\times10^4$         |
| 11  | $3.7\times10^4$         |
| 13  | $3.1\times10^4$         |
| 16  | $2.5\times10^4$         |
| 20  | $2.0\times10^4$         |
| 26  | $1.5\times10^4$         |
| 40  | $1.0\times10^4$         |

### Table 3: Index of determinacy for values of $s$ in experiment A via FAD

| $s$ | Index of determinacy |
|-----|----------------------|
| 1   | 0.760                |
| 8   | 0.745                |
| 9   | 0.748                |
| 10  | 0.750                |
| 11  | 0.752                |
| 13  | 0.754                |
| 16  | 0.758                |
| 20  | 0.752                |
| 26  | 0.748                |
| 40  | 0.744                |

---

Fig. 4 Tendency of $H_2$ for dataset partitions in the experiment A
always positive and the greatest value are obtained for \( s = 16 \). These considerations allow to apply the MFAD algorithm to a VL dataset not loadable entirely in memory to which the FAD algorithm is not applicable.

Now we compare the results obtained by using the MFAD method with the ones obtained by applying the SVR and MLP algorithms. For the comparison tests we have used the machine learning tool Weka 3.8.

In order to perform the tests by using the SVR algorithm, we repeat each experiment using the following different kernel functions: linear, polynomial, Pearson VII universal kernel, and Radial Basis Function kernel, and varying the complexity C parameter in a range between 0 and 10. To compare the performances of the SVR and MFAD algorithms we measure the index of determinacy and store it in every experiment.

In Fig. 9 we show the trend of the difference between the max values of \( r_{cl}^2 \) in SVR and MFAD.

Figure 9 shows that the difference between the optimal value \( r_{cl}^2 \) in SVR and MFAD is always under 0.02. In the comparison tests performed by using the MLP algorithm, we vary the learning rate and the momentum parameter in [0.1, 1]. We use a single hidden layer varying the number of nodes between 2 and 8. Furthermore, we set the number of epochs to 500 and the percentage size of validation set to 0.

In Fig. 10 we show the trend of the difference between the max value of \( r_{cl}^2 \) in MLP and MFAD.

Figure 10 shows that the difference between the max value of the index of determinacy in MLP and MFAD is under the value 0.016.

These results show that the MFAD algorithm of attribute dependency in massive datasets has comparable performances with the SVR and MLP nonlinear regression algorithms. Moreover, it has the advantage of having a smaller number of parameters compared to the other two algorithms, therefore it has greater usability and can be easily integrated into expert systems and intelligent systems for the analysis of dependencies between attributes in massive datasets. Indeed, the only two parameters for the execution of the MFAD algorithm are the number of subsets and the threshold value of the index of determinacy.

| Table 4 | Index of determinacy for values of \( s \) in experiment B via FAD | Table 5 | Index of determinacy for values of \( s \) in the experiment C via FAD |
|---------|-------------------|---------|-------------------|
| \( s \) | Index of determinacy | \( s \) | Index of determinacy |
| 1       | 0.881             | 1       | 0.785             |
| 8       | 0.872             | 8       | 0.776             |
| 9       | 0.872             | 9       | 0.776             |
| 10      | 0.874             | 10      | 0.778             |
| 11      | 0.875             | 11      | 0.780             |
| 13      | 0.877             | 13      | 0.781             |
| 16      | 0.878             | 16      | 0.781             |
| 20      | 0.878             | 20      | 0.780             |
| 26      | 0.875             | 26      | 0.779             |
| 40      | 0.872             | 40      | 0.777             |
The FAD method presented in (Martino et al. 2010a) can be used as a regression model for finding attribute dependencies in datasets: the inverse multiple F-transform can approximate the regression function. But this method can be expensive for massive datasets and for VL datasets not loaded in memory. Then we propose a variation of the FAD method for massive datasets called MFAD: The dataset is partitioned in s subsets equally sized, to each subset the FAD method is applied by calculating the inverse F-transform. Approximated by a weighted mean where the weights are given from the index of determinacy assigned to each subset. For testing the performance of the MFAD method, we compare tests with respect to the FAD method on an L dataset of the ISTAT 2011 census data. The results show that the performances obtained in MFAD are well comparable in FAD. The comparison tests show that the MFAD algorithm has performances comparable with SVR and MLP algorithms, moreover it has greater usability due to the lower number of parameters to be selected.

These results allow us to conclude that MFAD provides acceptable performance in the detection of attribute dependencies in the presence of massive datasets. Therefore, unlike FAD, MFAD can be applied to massive data and can represent a trade-off between usability and high performance in detecting attribute dependencies in massive datasets.

The critical point of the algorithm is the choice of the number of subsets and the threshold value of the index of determinacy. Further studies on massive datasets are necessary to analyze if the choice of the optimal values of these two parameters depend on the type of dataset analyzed. Furthermore, we intend to experiment the MFAD algorithm in future robust frameworks such as expert systems and decision support systems.

Author contributions All authors contributed to the study conception and design. All authors contributed to material preparation, data collection and analysis. All authors wrote the first draft of the manuscript commented on previous versions of the manuscript. All authors read and approved the final manuscript.
Funding Open access funding provided by Università degli Studi di Napoli Federico II within the CRUI-CARE Agreement. This research received no external funding.

Declarations

Conflict of interest The authors declare no conflict of interest.

Ethical approval This research does not contain any studies involving human participants performed by any of the authors.

Informed consent Informed consent was obtained from all individual participants included in the study.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Anguita A, Ridella S, Rivieccio F (2005) K-fold generalization capability assessment for support vector classifiers. In: Proceedings of the IEEE international joint conference on neural networks, ICNN 2005, pp. 855–858. https://doi.org/10.1109/ IJCNN.2005.1555964

Chen CLP, Zhang CY (2014) Data-intensive applications, challenges, techniques and technologies: a survey on big data. Inf Sci 275:314–347. https://doi.org/10.1016/j.ins.2014.01.015

Chen C, Li K, Duan M, Li K (2017) Chapter 6—extreme learning machine and its applications in big data processing. In: Big data analytics for sensor-network collected intelligence, data-centric systems, pp. 117–150. https://doi.org/10.1016/ B9780128093931.000064.

Cheng CH, Tan P, Jin R (2010) Efficient algorithm for localized support vector machine. IEEE Trans Knowl Data Eng 22(4):537–549. https://doi.org/10.1109/TKDE.2009.116

Collobert R, Bengio S (2004) Links between perceptrons, MLPs and SVMs. In: ICML ’04: proceedings of the 21st international conference on machine learning. https://doi.org/10.1145/ 1015330.1015415

Cybenko G (1989) Approximation by superpositions of a sigmoidal function. Math Control Signal Syst 2:303–314. https://doi.org/10.1007/BF02551274

Dean J (2014) Big Data, data mining, and machine learning: value creation for business leaders and practitioners. Wiley & Sons Inc., New York. ISBN: 9780470477185

Di Martino F, Sessa S (2007) Compression and decompression of image with discrete fuzzy transforms. Inf Sci 177:2349–2362. https://doi.org/10.1016/j.ins.2006.12.027

Di Martino F, Sessa S (2012) Fragile watermarking tamper detection with images compressed by fuzzy transform. Inf Sci 195:62–90. https://doi.org/10.1016/j.ins.2012.01.014

Di Martino F, Loia V, Perfilieva I, Sessa S (2008) An image coding/decoding method based on direct and inverse fuzzy transforms. Int J Approx Reason 48:110–131. https://doi.org/10.1016/j.ijar. 2007.06.008

Di Martino F, Loia V, Sessa S (2010a) Fuzzy transforms and attribute dependency in data analysis. Inf Sci 180:493–505. https://doi.org/10.1016/j.ins.2009.10.012

Di Martino F, Loia V, Sessa S (2010b) Fuzzy transforms for compression and decompression of color videos. Inf Sci 180:3914–3931. https://doi.org/10.1016/j.ins.2010.06.030

Di Martino F, Loia V, Sessa S (2011a) Fuzzy transforms method in prediction data analysis. Fuzzy Sets Syst 180:146–163. https://doi.org/10.1016/j.fss.2011.09.009

Di Martino F, Loia V, Sessa S (2011b) A segmentation method for images compressed by fuzzy transforms. Fuzzy Sets Syst 161:56–74. https://doi.org/10.1016/j.fss.2009.08.002

Draper NR, Smith H (1988) Applied regression analysis, Wiley & Sons Inc., New York. ISBN: 9780471170822

Drucker H, Burges CJC, Kaufman L, Smola AJ, Vapnik V (1996) Support vector regression machines. In: NIPS’96 proceedings of the 9th international conference on neural information processing systems 1996, pp. 155–161. MIT Press

Han H, Jian X (2019) Overcome support vector machine diagnosis overfitting. Cancer Inform 13(1):145–158. https://doi.org/10.4137/CIN.S13875

Han M, Kamber M, Pei J (2012) Data mining: concepts and techniques, 3rd ed. Morgan Kaufmann (Elsevier). ISBN: 9780123814791

Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning: data mining, inference, and prediction. Springer, New York. https://doi.org/10.1007/9780387848587

Haykin S (1999) Neural networks: a comprehensive foundation, 2nd ed. Prentice Hall. ISBN: 0132733501

Haykin S (2009) Neural networks and learning machines, 3rd ed. Prentice Hall. ISBN: 100131471392

Johnson RA, Wichern DW (1992) Applied multivariate statistical analysis, Prentice-Hall International, London. ISBN: 9780131877153

Jun S, Lee SJ, Ryu JB (2015) A divided regression analysis for big data. Int J Softw Eng Appl 9(5):21–32. https://doi.org/10.14257/ ijseia.2015.9.5.03

Lee YS, Yen SJ (2004) Classification based on attribute dependency. In: Proceedings of 6th international conference DaWaK’ 04. Lecture Notes in Computer Sciences, 3192:259–268. ISBN: 9783540487604

Leskovec J, Rajaraman A, Ullmann JD (2014) Mining of massive datasets, Cambridge University Press, 2nd ed. ISBN: 9781107077232

Mitra S, Pal SK, Mitra P (2002) Data mining in soft computing framework: a survey. IEEE Trans Neural Netw 13(1):3–14. https://doi.org/10.1109/TNN.2001.977258

Murtagh F (1991) Multilayer perceptrons for classification and regression. Neurocomputing 2(3–5):183–197. https://doi.org/10.1016/ 0925-2312(91)900235

Peng H, Choi D, Liang C (2013) Evaluating parallel logistic regression systems 1996, pp. 155–161. MIT Press

Perfilieva I (2006) Fuzzy transforms: theory and applications. Fuzzy Sets Syst 157:993–1023. https://doi.org/10.1016/j.fss.2005.11.012

Perfilieva I, Novák V, Dvovář A (2008) Fuzzy transforms in the analysis of data. Int J Approx Reason 48:36–46. https://doi.org/ 10.1016/j.ijar.2007.06.003

Piatecky-Shapiro G, Frawley WJ (1991) Knowledge discovery in databases. Cambridge (MA), MIT Press. ISBN: 9780262660709
Raju KS, Murti MR, Rao MV, Satapathy SC (2018) Support vector machine with K-fold cross validation model for software fault prediction. Int J Pure Appl Math 118(20):331–334
Schmidhuber J (2014) Deep learning in neural networks: an overview. Neural Netw 61:85–117. https://doi.org/10.1016/j.neunet.2014.09.003
Segata N, Blanzieri E (2009) Fast local support vector machines for large datasets. In: Perner P (ed) Machine learning and data mining in pattern recognition. MLDM 2009. Lecture notes in computer science, vol 5632. Springer, Berlin, pp 295–310. https://doi.org/10.1007/978-3-642-03070-3_22
Singh S, Firdaus T, Sharma AK (2015) Survey on big data using data mining. Int J Eng Dev Res 3(4):135–143
Tanaka H (1987) Fuzzy data analysis by possibilistic linear models. Fuzzy Sets Syst 24:363–375. https://doi.org/10.1016/01650114(87)90039
Thomas P, Suhner MC (2015) A new multilayer perceptron pruning algorithm for classification and regression application. Neural Process Lett 42(2):437–458. https://doi.org/10.1007/s11063-014-93665
Vucetic M, Hudac M, Vujošević M (2013) A new method for computing fuzzy functional dependencies in relational database systems. Expert Syst Appl 40(7):2738–2745. https://doi.org/10.1016/j.eswa.2012.11.019
Wood SN, Goude Y, Shaw S (2015) Generalized additive models for large data sets. J R Stat Soc Ser C (Appl Stat) 64(1):139–155. https://doi.org/10.1111/rssc.12068
Wu X, Zhu X, Wu GQ, Ding W (2014) Data mining with Big Data. IEEE Trans Knowl Data Eng 26(1):97–107. https://doi.org/10.1109/TKDE.2013.109
Yao L, Ge Z (2019) Distributed parallel deep learning of hierarchical extreme learning Machine for multimode quality prediction with big process data. Eng Appl Artif Intell 81:450–465. https://doi.org/10.1016/j.engappai.2019.03.011
Yen SJ, Lee YS (2011) A neural network approach to discover attribute dependency for improving the performance of classification. Expert Syst Appl 38(10):12328–12338. https://doi.org/10.1016/j.eswa.2011.04.011
Zheng J, Shen F, Fan H, Zhao J (2013) An online incremental learning support vector machine for large-scale data. Neural Comput Appl 22(5):1023–1035. https://doi.org/10.1007/s0052101107931

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.