sensors

MDPI

Article

Feature Selection for Health Care Costs Prediction Using Weighted Evidential Regression †

Belisario Panay [1,*], Nelson Baloian [1,*], José A. Pino [1], Sergio Peñafiel [1], Horacio Sanson [2] and Nicolas Bersano [2]

[1] Department of Computer Science, Universidad de Chile, Santiago 8320000, Chile; jpino@dcc.uchile.cl (J.A.P.); spenafie@dcc.uchile.cl (S.P.)
[2] Allm Inc., Tokyo 150-0002, Japan; horacio@allm.inc (H.S.); n.bersano@allm.inc (N.B.)
* Correspondence: bpanay@dcc.uchile.cl (B.P.); nbaloian@dcc.uchile.cl (N.B.)
† Presented at the 13th International Conference on Ubiquitous Computing and Ambient Intelligence UCAmI 2019, Toledo, Spain, 2–5 December 2019.

Received: 30 June 2020; Accepted: 3 August 2020; Published: 6 August 2020

Abstract: Although many authors have highlighted the importance of predicting people's health costs to improve healthcare budget management, most of them do not address the frequent need to know the reasons behind this prediction, i.e., knowing the factors that influence this prediction. This knowledge allows avoiding arbitrariness or people's discrimination. However, many times the black box methods (that is, those that do not allow this analysis, e.g., methods based on deep learning techniques) are more accurate than those that allow an interpretation of the results. For this reason, in this work, we intend to develop a method that can achieve similar returns as those obtained with black box methods for the problem of predicting health costs, but at the same time it allows the interpretation of the results. This interpretable regression method is based on the Dempster-Shafer theory using Evidential Regression (EVREG) and a discount function based on the contribution of each dimension. The method "learns" the optimal weights for each feature using a gradient descent technique. The method also uses the nearest k-neighbor algorithm to accelerate calculations. It is possible to select the most relevant features for predicting a patient's health care costs using this approach and the transparency of the Evidential Regression model. We can obtain a reason for a prediction with a k-NN approach. We used the Japanese health records at Tsuyama Chuo Hospital to test our method, which included medical examinations, test results, and billing information from 2013 to 2018. We compared our model to methods based on an Artificial Neural Network, Gradient Boosting, Regression Tree and Weighted k-Nearest Neighbors. Our results showed that our transparent model performed like the Artificial Neural Network and Gradient Boosting with an R^2 of 0.44.

Keywords: health care costs; dempster–shafer theory; supervised learning; regression; feature selection; evidential regression; interpretable prediction

1. Introduction

Health care expenditure is one of the most critical issues in today's society. World Health Organization (WHO) statistics show that global health care expenditure was approximately US$ 7.5 trillion, equivalent to 10% of the global GDP in 2016 [1]. One of the reasons for these high expenses in care is the low accountability in health care in some countries, for example, inefficiencies in the US health care system

result in unnecessary waste, provoking a large discrepancy between spending and returns in care [2]. If we could predict health care costs for each patient with high certainty, problems such as accountability could be solved, enabling control over all parties involved in patients' care. It could also be used for other applications such as risk assessment in the health insurance business, allowing competitive insurance premiums, or as input information for developing new government policies to improve public health.

With the current frequently used electronic health records (EHRs), an interest has emerged in solving accountability problems using data mining techniques [3]. There have been various approaches to predict health care costs for large groups of people [4,5]. On the contrary, prediction for an individual patient has rarely been tackled. Initially, rule-based methods [6] were used for trying to solve these problems requiring domain knowledge as if-then rules. The downside of this method is the requirement of a domain expert to create the rules, thus making the solution expensive and limited to the dataset being used. In the current state-of-the-art, statistical and supervised learning methods are preferred with the latter getting the best performance. The reason for best performance is the skewed and heavy right-hand tail with a spike at zero present in the distribution of health care costs [7].

Supervised learning methods can be evaluated by performance and interpretability; usually, the most sophisticated methods are the ones that have best performance, sacrificing interpretation (e.g., Random Forest, Artificial Neural Networks, and Gradient Boosting). A drawback of these high performing machine learning algorithms in health care is their black-box nature, especially in critical use cases. Even though health care cost prediction is not a critical use case, using patients' personal and clinical information for this problem could suffer biased results without an interpretable method. Interpretable methods would allow patients, physicians, and insurers to understand the rationale behind a prediction, giving them the option to accept or reject the knowledge the method is providing.

The aim of this work is to create an intrinsic interpretable regression method for health care costs prediction, with a performance comparable to state-of-the-art methods, inspired by the work of Peñafiel et al. [8], where an interpretable classifier based on the Dempster–Shafer Theory (DST) [9] was presented. The Dempster–Shafer Theory (DST) [9] is a generalization of the Bayesian theory. It is more expressive than classical Bayesian models since it allows us to assign "masses" to multiple outcomes measuring the degree of uncertainty of the process. We could have extended the work of the interpretable classifier but as it is a pure classification algorithm, the output is assumed to be discrete, and some procedures do not apply to continuous outputs, e.g., the time complexity grows exponentially with the number of classes. Petit-Renaud and Denœux [10] introduced the use of Dempster–Shafer theory for regression problems; they presented a regression model that uses DST called Evidential Regression (EVREG) to find the expected value of a variable using a set of examples as evidence. We will extend EVREG using a weighted distance and gradient descent, which enables the model to do feature selection (FS) tasks and use it in the health care costs prediction problem. The weight of each feature will represent the importance of this feature to predict an outcome in a dataset; thus, a set of masses with these weights will be computed; the masses will represent the importance of samples in the training set to predict an outcome.

Previous works in health care costs prediction [11–13] have reached to the conclusion that clinical features yield the same performance as using only cost predictors without the proper FS experiment. One aim of our work is to prove these claims with a proper FS method. Our research question is whether it is possible to develop an interpretable method with FS capabilities that has a similar performance to black-box methods for the health care cost prediction problem. We used Japanese health records from Tsuyama Chuo Hospital to test our answer. These records include medical checkups, exam results, and billing information from 2013 to 2018. We used them to compare our method performance with the performance obtained by less interpretable methods such as Artificial Neural Networks (ANNs) and Gradient boosting (GB).

First, we will describe the state-of-the-art for health care cost predictions, introduce FS and DST. Then we will describe our proposed model, which we named Weighted Evidential Regression (WEVREG) with its implementations and improvements. Finally, we will present the model performance in feature selection tasks against prediction methods with FS capabilities using synthetic data, besides its performance in health care costs prediction.

Our results show that WEVREG is able to perform FS tasks obtaining better performance for features with complex dependencies. In health care costs prediction, our method outperforms ANN, and gets similar results to GB but cannot reach the same performance. The features selected by our method show that cost variables are the most important ones to predict future costs, confirming the results obtained in previous works.

This paper extends the work presented previously in [14] by improving the prediction algorithm and showing how this model can perform feature selection. The paper also explains which are the features in the input data that are most important for making the prediction of the health costs.

2. Related Work

2.1. Health Care Cost Prediction

Health care costs commonly have a spike at zero and a skewed distribution with a heavy right-hand tail; statistical methods (e.g., linear regression) suffer from this characteristic in small to medium sample sizes [15]. Advanced methods have been proposed to address this problem, for example, Generalized Linear Models (GLMs) where a mean function (between the linear predictor and the mean) and a variance function (between the mean and variance on the original scale) are specified and the parameters are estimated given these structural assumptions [16]. Another example is the two-part and hurdle model, where a Logit or Probit model is used in the first instance to estimate the probability of the cost being zero, and then if it is not, a statistical model is applied, such as log-linear [17] or GLM. The most complex statistical method used to solve this problem is the Markov chain model; an approach based on a finite Markov chain suggested estimating resource use over different phases of health care [18]. Mihaylova et al. [19] present a detailed comparison of statistical methods in health care cost prediction.

Supervised learning methods have been extensively used to predict health care costs; the data used for these methods vary. While a few works use only demographic and clinical information (e.g., diagnosis groups, number of admissions and number of laboratory tests) [20], the majority have incorporated cost inputs (e.g., previous total costs, previous medication costs) as well [11–13,21], obtaining better performance. GB [22] excels as the method with the best performance for this problem [13], which is an ensemble-learning algorithm, where the final model is an ensemble of weak regression tree models, which are built in a forward stage-wise fashion. The most essential attribute of the algorithm is that it combines the models by allowing optimization of an arbitrary loss function, in other words, each regression tree is fitted on the negative gradient of the given loss function, which is set to the least absolute deviation [23]. ANNs come close to the performance of GB. An ANN is an extensive collection of processing units (i.e., neurons), where each unit is connected with many others; ANNs typically consist of multiple layers, and some goal is to solve problems in the same way that the human brain would do it [24]. Another type of model with good results is the M5 Tree [12]; this algorithm is also a Regression Tree (RT), where a Linear Regression Model is used for building the model and calculating the sum of errors as opposed to the mean [25].

Most health care expenses of a population are generated by a small group, as Bertsimas et al. [11] showed in their dataset: 80% of the overall cost of the population originates from only 20% of the most expensive members. Therefore, a classification phase is suggested to classify patients in a risk bucket when trying to improve the performance of the methods listed above. Morid et al. [7] reported that for low-risk buckets, GB obtains the best results, but for higher ones, ANN is recommended. It has also been

found that costs rise sharply with nearness to death [26–28]. This fact needs to be taken into account when building the embedding for the input vectors of a health care costs dataset.

2.2. Feature Selection

Nowadays, there are several datasets with high dimensionality. Reducing the number of features while maintaining the models performance has become indispensable. This strategy decreases the dimensionality of the Euclidean space, therefore preventing the curse of the dimensionality phenomenon that causes a drop on models accuracy [29]. Feature selection (FS) is the ability to select the features with the highest correlation to the target variable [30]; in particular they select a subset of features based on a certain evaluation criteria. Some features that do not meet the criteria are eliminated; the goal is to eliminate redundant or irrelevant features that are a priori unknown. This strategy decreases the number of dimensions, often resulting in an increase of models accuracy and time performance. FS has been of great importance in areas such as DNA microarray analysis, text categorization and information retrieval [31,32]. FS methods can be classified in three types: filter, wrapper and embedded methods [33]. Filter methods are typically the simplest and cheapest to compute; they apply statistical techniques to test the correlation of each feature with the target variable: normally a threshold is set to choose the most relevant features. The most well-known filter methods include Pearson's correlation coefficients [34] which assigns a value to two variables, between -1 and 1 representing the linear dependency between them. F-score [35] is another example, where the weighted harmonic mean of the test precision and recall is computed varying from 0 to 1. The disadvantage of these methods is that they cannot detect complex relations as they do not reveal mutual information between features. Wrapper methods are expensive because they test multiple subsets of features in a prediction model and select the subset maximizing the performance of the prediction. The most well-known method for regression problems is the Linear Support Vector Regressor (L1-SVR) [36] which given a toleration error, finds the hyper-plane that maximizes the margin between vectors. Feature importance is then obtained from the SVR coefficients. Embedded methods, unlike the previous methods, include the feature selection process as their learning phase. Common embedded methods include various types of decision tree algorithms. Some of the most popular ones are the Random Forest (RF) [37]—a method which uses multiple decision trees to reach an outcome—and Gradient Boosting (GB) [38], where weak decision trees are built, and features are selected sparsely following an important change in the impurity function.

Since FS is an important pre-processing step in most machine learning applications, it has been widely studied with new methods constantly appearing. There has been recent interest in FS algorithms based on k-neighbors. ReleifF [39] is a filtering method that randomly samples an instance of the data and locates its k nearest neighbor, the k instances are then used to update the score of each feature. Other recent approaches, Regression Gradient Guided Feature Selection (RGS) [40] and Weighted Nearest Neighbors(WkNN) [41] are methods that use a Weighted k-NN model with a gradient descent as an optimization approach to find the optimal weight vector used in the k-NN distance function. These two algorithms differ in the gradient descent algorithm and discount function used to find the optimal importance of each feature.

2.3. Dempster–Shafer Theory

Let X be the set of all states of a system called frame of discernment. A mass assignment function m is a function satisfying:

$$m : 2^X \rightarrow [0,1], \quad m(\phi) = 0, \quad \sum_{A \subseteq X} m(A) = 1 \tag{1}$$

The term $m(A)$ can be interpreted as the probability of getting precisely the outcome A, and not a subset of A.

Multiple evidence sources expressed by their mass assignment functions of the same frame of discernment can be combined using the Dempster Rule (DR) [42]. Given two mass assignment functions m_1 and m_2, a new mass assignment function m_c can be constructed by the combination of the other two using the following formula:

$$\begin{aligned} m_c(A) &= m_1(A) \oplus m_2(A) \\ &= \frac{1}{1-K} \sum_{B \cap C = A \neq \phi} m_1(B) m_2(C) \end{aligned} \tag{2}$$

where K is a constant representing the degree of conflict between m_1 and m_2 and is given by the following expression:

$$K = \sum_{B \cap C = \phi} m_1(B) m_2(C). \tag{3}$$

Petit-Renaud and Denœux [10] introduced a regression analysis algorithm based on a fuzzy extension of belief functions, called evidence regression (EVREG). Given an input vector **x**, they predict a target variable **y** in the form of a collection of evidence associated with a mass of belief. This evidence can be fuzzy sets, numbers, or intervals, which are obtained from a training set based on a discount function that takes their distance to the input vector **x** and is pooled using the Dempster combination rule (Equation (2)). They showed that their methods work better than similar standard regression techniques such as the Nearest Neighbors using data of a simulated impact of a motorcycle with an obstacle.

The EVREG model has been used for predicting the time at which a system or a component will no longer perform its intended function (machinery prognostic) for industrial application. Niu and Yang [43] used the EVREG model to construct time series, whose prediction results are validated using condition monitoring data of a methane compressor to predict the degradation trend. They compared the results of the EVREG model with six statistical indexes, resulting in a better performance of the EVREG model. Baraldi et al. [44] used the model to estimate the remaining useful life of the equipment. Their results have shown the effectiveness of the EVREG method for uncertainty treatment and its superiority over the Kernel Density Estimation and the Mean-Variance Estimation methods in terms of reliability and precision.

3. Data and Problem Description

The problem we address in this work is predicting future health care cost of individuals, using their past medical and cost information. This is a supervised learning problem, which can be formally specified as a regression problem where the input vector $x = (x_0, x_1, \ldots, x_n)$ is an individual's past medical and cost information and the target variable y is that person's health care expenses in a future period (e.g., a year).

The records used for this work were provided by Tsuyama Chuo Hospital, a Japanese hospital located in Okayama prefecture. These records were gathered between 2013 and 2018.

Japan has universal coverage for social health insurance; the system is composed of three sub-systems, National Health Insurance (self-employment), Society Health insurance (for employees) and a Special Scheme for the elderly (75 or older). Every citizen must join one of these three sub-systems according to their occupational status and age. The premium charge for each person is set by each insurer depending on the person's income. Each medical organization is paid by a fee-for-service principle; at the end of the month every medical facility in Japan has to send a set of claims to be reimbursed by an insurer (as a claim sheet); the insurers have the right to decline a claim if it is incorrect or seems unnecessary.

Medical facilities use a special software for the production of a claim sheet. This software registers all procedures, drugs and devices for each patient. Each procedure has a standard code set by the Ministry of Health, Labor and Welfare (MHLW) that can be translated directly to the International Statistical

Classification of Diseases and Related Health Problems codification (ICD-10) [45], a medical classification list created by the World Health Organization (WHO).

Every claim sheet send by health facilities all over Japan are gathered by the MHLW in a National Database [46]. The database contains a detailed information on patients such as provided service, age, sex, date of consultation, date of admission, date of discharge, procedures and drugs provided with volumes and tariffs. 1700 million records are registered annually. Unfortunately, we do not have access to the National Database, but we have access to the electronic claims (claim sheets) sent by Tsuyama Chuo hospital to the National Database between 2013 and 2018.

The claims are stored in a set of files; we had to transform these files to study the data. The format of the claims file is confusing without previous knowledge of the structure. The detailed documentation of this format can be found at the Medical Remuneration Service website (http://www.iryohoken.go.jp/shinryohoshu/file/spec/22bt1_1_kiroku.pdf). A short summary of the file format is shown in Table 1.

Table 1. Insurance claims.

Header	Name	Description
IR	Medical institution	Details of the medical institution.
RE	Insured details	Patient details with dates and demographics.
HO	Insurer details	Patient insurer information.
KO	Public expenses	Patient public expense information.
KH	Special information	Patient especial information (free text).
SY	Diagnosis	Patient diagnosis in MHLW coding.
SI	Procedure	Details for a patient treatment.
IY	Medications	Details for the medicines given.
TO	Specific equipment	Specific equipment details used in a patient.
CO	Comment	Comments for diagnoses or symptoms (free text).
SJ	Symptoms	Patients symptoms.

Since a patient's data are dispersed within these files, we could not use these raw data to predict the health care expenditure of patients. We used the data in these files to create a patient's representation that we could use for the prediction of an individual's health care costs. Each patient in the insurance claims could be identified by a unique identifier, which enabled us to follow all patients throughout the years.

We had access to patients' health checkups as well as to the data sent by Tsuyama Chuo hospital to the National Database. Every Japanese worker needs to take a yearly health checkup to start or continue working at a company, so there were many patients with health checkup data.

Our dataset had patients' monthly history between 2013 and 2018. However, there were many missing values because most patients had few claims each year. Therefore, we grouped claims yearly so that we could have fewer missing values. We created three different scenarios for this purpose as follows. **Scenario 1**: We used a single year of history to predict the next cost value. **Scenario 2**: We used two years of data to predict the third one, and **Scenario 3**: We used five years of history to predict the costs of the sixth year. We filtered out patients in each scenario if they did not have the required history available, and thus, the sets shrank in each scenario. In the case of missing health check values, we opted to use the median of the exam to replace null values. Table 2 shows the basic statistics of the sets in each scenario.

Table 2. Statistics of patients' records in each scenario.

Statistics	Scenario 1	Scenario 2	Scenario 3
Total number of patients	71,001	33,646	8,810
Mean costs	11,030	11,536	12,420
Mean age	54.00	58.00	63.00
% Male	48.81	48.54	48.56
% Female	51.19	51.46	51.44

4. Proposed Model

A regression task predicts the value of a target variable y using as input an arbitrary vector x. For a regression method to succeed at this task, the predicted target variable value $\hat{y}$ needs to be as close as possible to the real target variable y value. In particular, in a Supervised Learning problem, variable $\hat{y}$ is deduced from a training set or the set of examples that are taken from past observations. In summary a regression task solves the problem of finding a function $f(x)$ which best explains the target variable y.

Petit-Renaud and Denœux presented a regression method based on the Dempster–Shafer Theory (DST) called Evidential Regression (EVREG) [10]. The EVREG model uses a set of observations that have occurred in the past as evidence in order to predict the target value of a new observation. Each observation in this evidence set is given a mass which represents the similarity of the new observation with each observation in the set. We calculate this similarity using a distance function (e.g., Euclidean distance) in the feature space of the observations. The DST ensures that the masses of this evidence set add up to 1, so they can be easily transformed to a probability distribution. Then an expected value is computed as the mass of each past observation m_i, times its observed target value (y_i), as shown in Equation (4).

$$E[y] = \sum_{i=1}^{N} m_i * y_i \tag{4}$$

DST is characterized for reasoning with uncertainty. EVREG uses the width of the observed target variable interval (difference between maximum and minimum target variables) to represent uncertainty as another piece of evidence. The importance given by the model to this interval represents the uncertainty the model has when predicting an outcome. For example, when the new observation is at a great distance from the evidence set, the model assigns a high value to the evidence of the variable y interval; thus we get a high uncertainty in the model outcome, resulting in upper and lower bounds for the predicted target variable proportional to the interval of the target variable observed in the evidence set.

EVREG has been used for predicting the time at which a system or a component will no longer perform its intended function (machinery prognosis) for industrial applications [43,44]. Niu and Yang [43] used the EVREG in a time series task, to predict the degradation trend of a methane compressor. They compared the results of the EVREG model with six statistical indexes; the results showed the EVREG model had the best performance. Baraldi et al. [44] used the model to estimate the remaining useful life of industrial equipment. Their results have shown the effectiveness of the EVREG method for uncertainty treatment and its superiority over the Kernel Density Estimation and the Mean-Variance Estimation methods in terms of reliability and precision.

We will first describe EVREG (Section 4.1) in this section. Then, its time complexity and time improvement will be explained in Section 4.2. We will extend EVREG using gradient descent and a weighted distance function in Sections 4.3 and 4.4; the goal will be to improve the model regression performance and enable it to perform feature selection tasks, which will allow it to rank features in a space. Using this new distance function and optimization approach, the model will update the weight of each

dimension. The extended model will be able to rule out or assign low weights to irrelevant features, and assign higher values to the ones that strongly influence the output. This is an important characteristic since feature selection has been shown to improve model accuracy and computing times [40,41,47]. Our aim is that this enhanced EVREG method will perform as well as state-of-the-art regressors for tabular and time series regression problems. In addition, the model will possess a feature selection ability, ranking the features representing the correlation of each feature with the target variable, enabling its users to gain more knowledge of the used data.

4.1. Evidential Regression

EVREG [10] is a Supervised Learning algorithm. This kind of algorithm uses a sample set as examples to make a prediction. The samples in a Supervised Learning problem are called the training set. Formally, we define the training set as:

$$\mathcal{L} : \{e_i = (x_i, y_i)\}_{i=1}^{N} \tag{5}$$

where e_i is an element of the training set, x_i is the input vector of e_i and y_i its output or target value.

Let x be an arbitrary vector and y its corresponding unknown target variable. We need to derive the expected value $\hat{y}$ of variable y from the training set $\mathcal{L}$. Each element e_i of the training set is a piece of evidence concerning the value of y. The relevance of each element in $\mathcal{L}$ can be assumed to depend on the similarity between the arbitrary vector x and the input vectors x_i of e_i. It can be assumed that a suitable discount function that depends on a distance measure $\|\cdot\|$ can measure this similarity. If x is close to a vector x_i in $\mathcal{L}$, we assume that y is also close to y_i, which makes the element e_i an important piece of evidence. If x and x_i are at a great distance, it is safe to assume that y_i has a small effect on y, and provides only marginal information concerning y. We will use the Minkowski distance defined as:

$$d(x_i, x_j) = \Big(\sum_{k=1}^{l} |x_{ik} - x_{jk}|^p \Big)^{\frac{1}{p}} \tag{6}$$

where x_{ik} and x_{jk} are the values of vectors x_i and x_j at dimension k. When p is 1 we get the L1 or Manhattan distance and with p equal to 2 we get L2 or also known as the Euclidean distance. In the case of high dimensional spaces, the vectors become uniformly distant from each other, the ratio between the nearest and farthest vector approaches 1. This phenomenon is more present in the Euclidean than in the Manhattan distance metric [48,49], which makes the Manhattan distance to yield better results in distance-based algorithms in the presence of a high dimensional space. EVREG could use any distance measure (e.g., cosine distance) to represent the similarity between two vectors. We will use the Minkowski distance for its flexibility while testing the performance for values $p \in [1, 2]$. The value of p will be chosen as the one that yields the best prediction performance in each problem. We will define the discount function ϕ between vectors x_i and x_j using this distance measure as:

$$\phi(d(x_i, x_j)) = \exp\Big(-\frac{d(x_i, x_j)^2}{\gamma}\Big) \tag{7}$$

where ϕ is a decreasing function from $\mathbb{R}$ to $[0, 1]$ that needs to fulfill $\phi(0) \in [0, 1]$ and,

$$\lim_{d \to \infty} \phi(d) = 0 \tag{8}$$

Equation (7) is the well-known Radial Basis Function (RBF) that decreases monotonically with distance, commonly used as a kernel in the Support Vector Machine (SVM) classifier. Parameter γ is the radius of the function, intuitively γ defines how far the influence of a vector reaches. In Figure 1 we can

observe Equation (7) for different γ values. RBF-based methods are an active area of research [50,51]; various approaches exist to find the optimal parameters such as γ. This parameter can be learned using an optimization approach, but often the value is set manually by trial and error. We will try to find the optimal γ using a Grid Search approach in this work

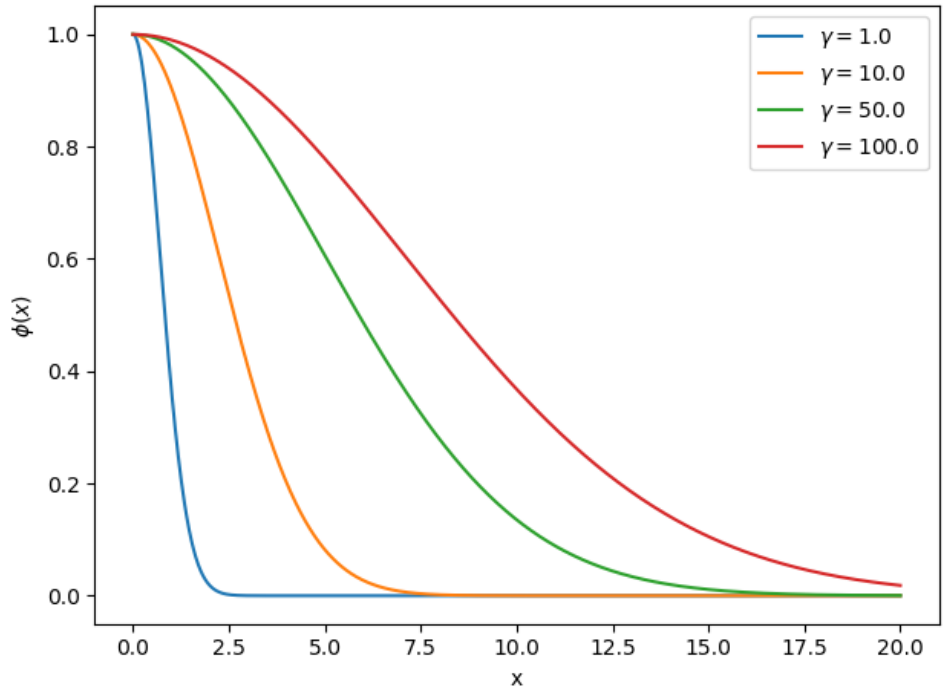


Figure 1. Radial Basis Function for different γ values.

The discount function ϕ will represent the similarity between two vectors (higher value means higher similarity); we can use this similarity function to compute the mass (influence) of each element in $\mathcal{L}$ given an arbitrary vector x using the Dempster rule of combination getting:

$$m_i(x) = \frac{1}{K}\phi(d(x, x_i)) \prod_{k!=i}(1 - \phi(d(x, x_k))) \tag{9}$$

where K is a normalization term defined by the DST as:

$$K = \prod_{j=1}^{N}(1 - \phi(d(x, x_i))) + \sum_{i=1}^{N}[\phi(d(x, x_i)) \prod_{k!=i}(1 - \phi(d(x, x_k)))] \tag{10}$$

In Equation (9), ϕ determines the influence of x_i in x and the product determines the effect of the evidence set between these two vectors. We can obtain the influence of every vector in set $\mathcal{L}$ using the previous formulas, but we need to consider one extra piece of evidence. We have observed the values of variable y in every example of set $\mathcal{L}$. EVREG takes this information into account. We will assume that variable y is bounded to the interval $[y_{inf}, y_{sup}]$, and for this interval we will define an additional mass called domain mass m^* calculated as:

$$m^*(x) = \frac{1}{K}\prod_{i=1}^{N}(1 - \phi(d(x, x_i))) \tag{11}$$

Since we assumed that y is bounded to the $[y_{inf}, y_{sup}]$ interval, there exists a probability density function $P(x)$ associated to $m_i(x)$ and $m^*(x)$. Smets et al. [52] showed that the Pignistic transformation could be used to transform the masses in EVREG to a probability function. In the particular case where output y is a real number, the Pignistic probability is defined as:

$$P(x) = \sum_{i=1}^{N} m_i(x) \cdot \delta_i(x) + \frac{m^*}{y_{sup} - y_{inf}} \tag{12}$$

Equation (12) is a mixture of Dirac distributions and a continuous uniform distribution. With this probability function, we can get the expected value of target variable y as the Pignistic expectation [10]:

$$\hat{y} = \sum_{i=1}^{N} m_i(x) \cdot y_i + \frac{m^*(x) \cdot (\sup_{y \in \mathcal{L}} y + \inf_{y \in \mathcal{L}} y)}{2} \tag{13}$$

where $\hat{y}$ is the expected or predicted value of target variable y. With the Pignistic expectation we have an upper and lower bound for variable $\hat{y}$ as:

$$\hat{y}^* = \sum_{i=1}^{N} m_i(x) \cdot y_j + m^*(x) \cdot \sup_{y \in \mathcal{L}} y \tag{14}$$

$$\hat{y}_* = \sum_{i=1}^{N} m_i(x) \cdot y_j + m^*(x) \cdot \inf_{y \in \mathcal{L}} y \tag{15}$$

We can observe that the interval $[\hat{y}_*, \hat{y}^*]$ contains variable $\hat{y}$. The width of this interval can be interpreted as the uncertainty of the response, which is directly associated with the mass of the domain of target variable y.

The computation time of a prediction grows quickly with the size of the training set. For a single prediction in training set $\mathcal{L}$ of size n, we need to compute the mass of each element e_i which has a vector x_i of dimension q. We start by pre-computing the discount function (ϕ) of the input vector x with every element e_i in the set $\mathcal{L}$. This computation takes $\mathcal{O}(q)$ time for each element in the set, taking a total time of $\mathcal{O}(nq)$ to compute each mass of $\mathcal{L}$ additionally to the discount function. In Equation (9) we need to compute a product sequence and the normalization term K. As we have pre-computed the discount functions for the training set, the product sequence takes only $\mathcal{O}(log(n))$. As for the normalization term K (Equation (17)) which takes most of the calculation time for computing Equation (9), the product sequence on the left side can also be computed in $\mathcal{O}(log(n))$. The sum of the right side takes time $\mathcal{O}(nlog(n))$ because of the product sequence contained in it. The domain mass also requires the normalization term K and the product sequence takes time $\mathcal{O}(nlog(n))$. Thus the time complexity for a single mass in the training set is $\mathcal{O}(nlog(n))$.

4.2. Improving Computing Time

All the masses of the training set need to be calculated to compute a single prediction. The mass calculation needs to be computed n times; since we compute a single mass in time $\mathcal{O}(nlog(n))$, the n masses will take time $\mathcal{O}(n^2log(n))$. Computing a prediction with the formula shown in Equation (13) uses the masses of the set $\mathcal{L}$, and the mass of the domain of target variables in the training set. This calculation requires to compute the maximum and minimum values, which needs time $\mathcal{O}(n)$, so we can get the prediction of input vector x in time $\mathcal{O}(n^2log(n))$.

The quadratic growth of each prediction makes it impossible for the EVREG model to work with large sets; therefore it is difficult to apply it for real-life problems. However, it has been suggested the use of a k-Nearest Neighbors (k-NN) approach [10] to improve computation time. Their results showed that a k-NN approach did not introduce a significant penalty to the algorithm performance. It is possible to create indexes for the k-NN search in $\mathcal{O}(n(q+k))$ with this approach in the following way. First, we compute the distance between x and x_i for each vector in the training set, then we iterate through the distances k times selecting the smallest distance. Now, only the masses of the k nearest neighbors have to be calculated when computing a new prediction of a vector x. The masses of samples that are not in the set of the nearest neighbors of x will be assumed to be null. In particular, we use a flat index implementation by Johnson et al. [53], for the exact nearest neighbor's search given its better execution times and memory usage compared to other existing solutions.

Given now that only the masses of the k Nearest Neighbors are relevant for a prediction, the mass of each sample in the training set for an input vector x is calculated as:

$$m_i(x) = \begin{cases} \frac{1}{K}\phi(d(x,x_i)) \prod\limits_{\substack{k!=i \\ x_k \in N(x)}} (1-\phi(d(x,x_h))) & \text{if } x_i \in N(x), \\ \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

where $N(x)$ is the set of k Nearest Neighbors of vector x. Now the normalization term K can be computed as:

$$K = \prod_{x_i \in N(x)} (1-\phi(d(x,x_i))) + \sum_{x_i \in N(x)} [\phi(d(x,x_i)) \prod_{\substack{h!=i \\ x_h \in N(x)}} (1-\phi(d(x,x_h)))] \tag{17}$$

Just the k neighbors are considered as evidence for a prediction of the domain mass, so the domain evidence is only attributed to its neighbors. We compute the calculation of the domain mass as:

$$m^*(x) = \frac{1}{K} \prod_{x_i \in N(x)} (1-\phi(d(x,x_i))) \tag{18}$$

Since now we compute only the masses of the k Nearest Neighbors, we have to change the size of the training set n by the number of neighbors k instead in the expression for computing the time complexity. In Equation (16) once we got the k Nearest Neighbors, the normalization term K will be computed in $\mathcal{O}(qklog(k))$. The discount function will still take $\mathcal{O}(q)$ and the product sequence will have $\mathcal{O}(qklog(k)$ complexity. Only the masses of the k neighbors will be needed to compute a prediction, so only k masses of the training set will be computed thus obtaining a complexity for a single prediction of $\mathcal{O}(qk^2log(k))$. Assuming that k is always significantly less than n, the conclusion is that the complexity for a prediction ends up being reduced to $\mathcal{O}(nqk)$ because of the complexity of the k-NN search.

We performed the following experiment to demonstrate the importance of the k-NN approach for speeding up the prediction computing time. We had two different settings, one for each approach: (i) the first approach used all the training set as evidence; we started with 1000 examples and ended at 30,000 with steps of size 500 in this case. (ii) the k-NN approach; the examples started at 100,000 and finished at 5,000,000 examples with steps of size 100,000. We fixed the number of neighbors to 10 and the number of dimensions used was only 1. The results are shown in Figure 2. Each value shown in the figure corresponds to the mean execution time of 100 experiments.

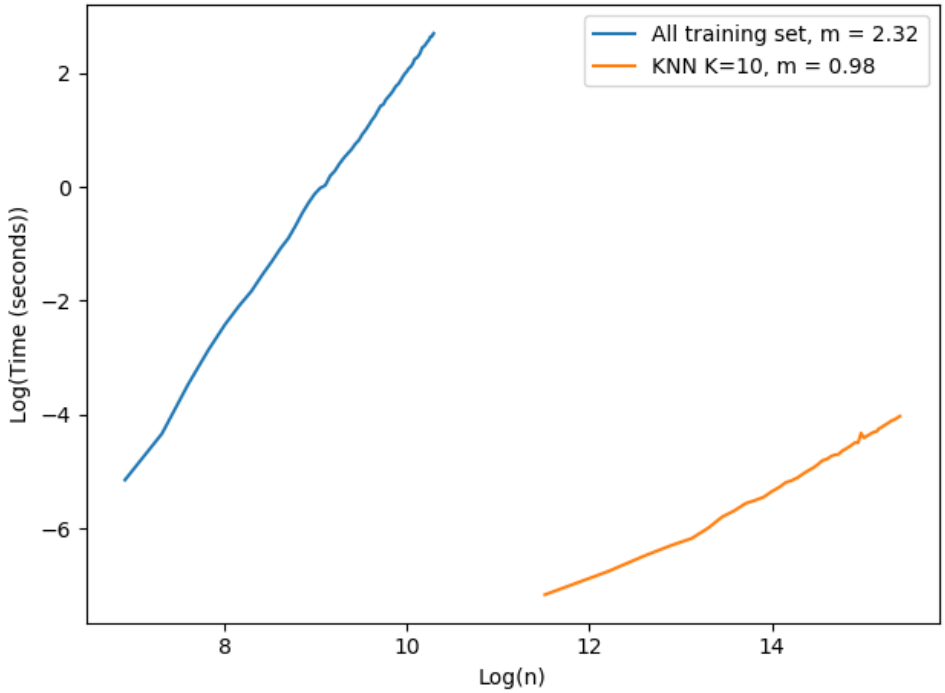


Figure 2. Time complexity single prediction.

The plot in Figure 2 is in a logarithmic scale. The slope of each curve was computed to observe the complexity of each approach. The x axis corresponds to the logarithmic number of samples in the training set, and the y axis is the logarithmic execution time of a single prediction in seconds. The number of vectors varied in each approach due to hardware constraints: case (i) we had to limit the number of vectors that could be tested because of excessive execution time and memory consumption. Case (ii) the machine where the experiment ran could not register execution times properly for a low number of vectors in the training set. As expected, the time complexity of the implemented EVREG with no optimization grew dramatically faster than the k-NN approach as seen by their slopes. The complexity of using all the training set approach seemed to have a larger time complexity than theoretically expected, as the slope in a complexity of $\mathcal{O}(n^2 log(n))$ was 2.12. In the k-NN approach we could easily observe a linearity with the size of the training set. Consequently, it became obvious that the k-NN approach significantly dropped execution times and made EVREG a viable option for real world problems.

4.3. Weighted Evidential Regression

This section proposes an improvement to the discount function used in EVREG based on ideas which has been previously introduced to enhance the well-known k-Nearest Neighbors Regressor (k-NN Regressor) [54], which is another regressor, similar to EVREG. The improved model will be called Weighted Evidential Regression (WEVREG) Model. The aim of this improvement is to boost prediction performance and allow the model to perform feature selection tasks. The k-NN Regressor is a simple and intuitive non-parametric regression method to estimate the output value of an unknown function for a given input. It "guesses" the output value using samples of known values as a training set, and computing the mean output value of the nearest neighbors of the input vector. All neighbors in this method are equally relevant for predicting the target variable in its original version. There is a variation of this algorithm

where the importance of each neighbor depends on a distance measure between them, thus using a weighted average of the k-NN vectors in the training set. This weighted k-NN Regressor is a kind of locally weighted regression [55]. The weight of each neighbor is proportional to its proximity to the input vector. The prediction is computed with this approach by the following expression:

$$\hat{f}(x) = \frac{1}{Z} \sum_{x' \in N(x)} f(x') \, \mathrm{e}^{\frac{-d(x,x')}{\beta}} \tag{19}$$

where $N(x)$ is the set of k Nearest Neighbors of vector x, $f(x')$ is the value of the target variable of neighbor x', $d(x, x')$ is a distance function between vector x and its neighbor x' (commonly the Euclidean distance), β is a parameter of the estimator and Z is a normalization function with value $Z = \sum_{x' \in N(x)} \mathrm{e}^{\frac{-d(x,x')}{\beta}}$.

Similar to the EVREG, the weighted k-NN regression uses the Euclidean distance and a Gaussian Radial Basis discount function to assign a weight to each one of its neighbors. Further improvements have been made to the weighted k-NN Regression based on the assumption that the target variable is most accurately predicted using only the most relevant features of the neighbors which adds a Feature Selection process to the algorithm. Navot et al. [40] introduced a weighted distance function for the weighted k-NN Regressor, which improved the model performance and enabled the model to perform a feature selection task. Given a weights vector w over the features of size q the distance function induced by w is defined as:

$$d(x_i, x_j) = \left(\sum_{k=1}^{l} |(x_{ik} - x_{jk}) \cdot w|^p \right)^{\frac{1}{p}} \tag{20}$$

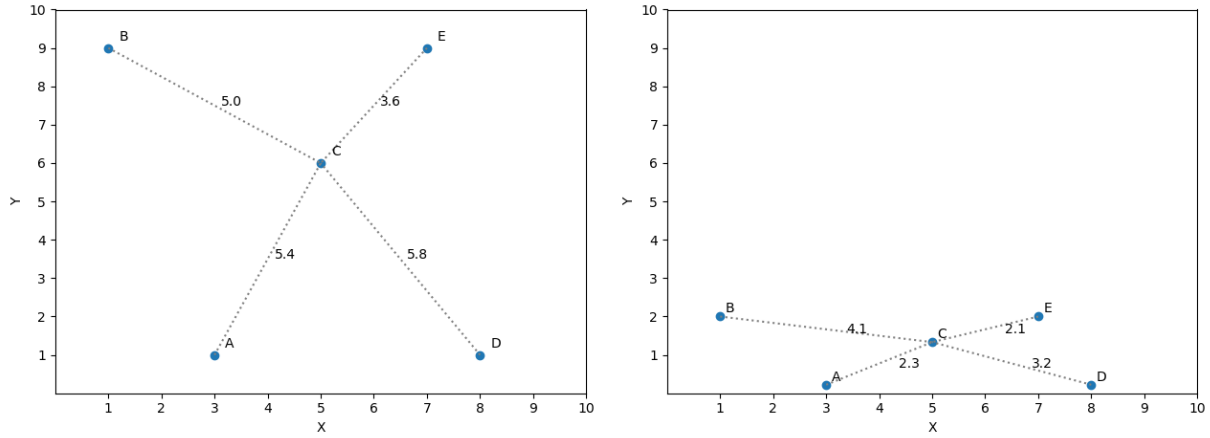
where input vector x_i and x_j have the same size q as the weights vector w. Each value of vector w represents the importance for each dimension in computing the distance between two vectors, i.e., the amount which a feature contributes to the distance between these two vectors. The model assumes that every dimension contributes the same to the distance between two vectors in EVREG Equation (6), and consequently, to our discount function (Equation (7)). However, this is not always the case; for example we could have a feature in our input that has no impact in the target variable we are trying to predict thus it contributes nothing to the similarity between those vectors. For instance, a patient age could have a great impact on that individual health care costs so to predict the costs we want to be closer to patients in the same age group, but maybe the eye color does not influence this cost, so we would not want to include two patients with the same eyes color just for this fact.

We will change the distance function in Equation (6) with the one that uses weights as presented in Equation (20) in order to improve the prediction performance of the EVREG model and to gain further understanding about the data.

We will also use the distance measure described in Equation (20), to compute the k-NN of each input vector x. The k-NN search implementation we use does not have a feature to use a custom distance measure to compute a k-NN search, so we will have to create the indexes with a transform training set space (applying the weights vector), and then transform the input vector x that will be predicted. We will get the same distance measure as described in Equation (20) with this operation.

A simple example is presented to ease the understanding of the chain of thoughts behind this idea. Let us assume we have a sample set of size 5 in a two dimensional space and we want to predict its third dimension. The input vectors (two dimensions) are shown in Figure 3a. The optimization discussed in Section 4.2 is used in this example, taking into account only the three closest neighbors to predict the target variables. If we want to predict the output value for vector C using the three closest neighbors, then we should consider A, B and E as its closest neighbors, shown in Figure 3a by the distance between C and all the vectors. Nevertheless, what if we find that the dimension y is not as significant as x for predicting

the target dimension? This observation means a variation in y does not affect in the same magnitude the prediction as a difference of the same size in x. The space to reflect the variation in magnitude is scaled, getting Figure 3b.



(**a**) Example input vectors.

(**b**) Transformed input vectors.

Figure 3. Feature transformation.

If we now compute the three closest neighbors for vector C then we get vector A, E and D, replacing vector B, thus reflecting the importance of feature x in the similarity function.

If we want the weight of a feature to represent the importance in an input vector, then the range of each one of the feature domains has to be of the same size. Otherwise, we could get a bigger weight of a dimension only to compensate the size of the domain. Therefore, it is recommended to normalize the input features (all values must be between 0 and 1) so that the weights are comparable among features.

We can find the optimal weight vector using an optimization approach such as gradient descent. We will describe the process of finding the optimal weights for a known training set in the next subsection.

4.4. Weight Learning

This subsection will present the process of finding the optimal weights vector in a known training set. We will use a gradient descent algorithm to find a vector w that minimizes the error of the prediction model.

The accuracy of a regressor is commonly measured by computing the difference between the predicted value $\hat{Y}$ and the actual value of the target variable Y. The estimation error is expressed by a loss function $L(Y, \hat{Y})$, $L : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$. Likewise in a Supervised Learning problem, our goal is to find a vector w that induces the smallest error in L using the training set $\mathcal{L}$ and a small generalization error in a validation set at the same time. We will use a gradient-based algorithm such as gradient descent with an Adam optimizer to get the optimal value for w, because of its clear mathematical justification for reaching optimum values [56].

The gradient descent algorithm is an iterative algorithm that optimizes variable values in order to find a function minimal value. We use the Mean Squared Error (MSE) as our error function [57], and we define the loss function L induced by w as:

$$L_w(Y, \hat{Y}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{21}$$

where n is the number of samples in the training set, Y is a vector of size n with the actual values for the target variable of each sample in the training set, and $\hat{Y}$ is the predicted value for each one of the samples in the training set. y and $\hat{y}_i$ are elements of Y and $\hat{Y}$ respectively.

After defining the function error L which is induced by the weights vector w, our goal is to find a vector w that yields the smallest error possible for the training set given. The estimator $\hat{w}$ for vector w is obtained by minimizing this criterion:

$$\hat{w} = argmin\ L_w(Y, \hat{Y}) \tag{22}$$

Since our loss function L_w is continuous and differentiable, we can use gradient descent algorithm to find $\hat{w}$. The gradient of L_w needs to be computed in every iteration of our algorithm in order to update the weights w, then w is updated by taking a step proportional to the negative of the gradient. We need the partial derivation of L_w for the gradient computation. This derivation is calculated as:

$$\frac{\partial L_w}{\partial w} = \frac{2}{n}\sum_{i=1}^{n}\frac{\partial(y_i - \hat{y}_i)}{\partial w} = \frac{2}{n}\sum_{i=1}^{n}\frac{\partial(y_i - \hat{y}_i)}{\partial \hat{y}_i}\cdot\frac{\partial \hat{y}_i}{\partial w} \tag{23}$$

We need to calculate the partial derivative of the predicted target variable $\hat{Y}$ with respect to w in order to solve Equation (23). Therefore, the calculation of the derivative of Equation (13) gives:

$$\frac{\partial \hat{y}_i}{\partial w} = \sum_{j=1}^{N}\frac{\partial m_j(x_i)}{\partial w}\cdot y_j + \frac{\partial m^*(x_i)}{\partial w}\cdot\frac{(\sup_{y\in\mathcal{L}} y + \inf_{y\in\mathcal{L}} y)}{2} \tag{24}$$

where the derivative of a single mass in set $\mathcal{L}$ is calculated as,

$$\frac{\partial m_j(x_i)}{\partial w} = \frac{K\frac{\partial \phi(x_i,x_j)}{\partial w} - \frac{\partial K}{\partial w}\cdot\phi(x_i,x_j)}{K^2}\cdot\prod_{k!=i}(1-\phi(d(x,x_k))) + \frac{\phi(x_i,x_j)}{K}\cdot\frac{\partial(\prod_{k!=i}(1-\phi(d(x,x_k))))}{\partial w} \tag{25}$$

With the derivatives of discount function ϕ and the normalization term K as,

$$\frac{\partial \phi(x_i,x_j)}{\partial w} = -\frac{2}{\gamma}\exp(\frac{-d(x_i,x_j)^2}{\gamma})\cdot d(x_i,x_j)\cdot\frac{\partial d(x_i,x_j)}{\partial w} \tag{26}$$

$$\frac{\partial K}{\partial w} = \frac{\partial(\prod_{j=1}^{N}(1-\phi(d(x,x_i))))}{\partial w} + \sum_{i=1}^{N}\frac{\partial[\phi(d(x,x_i))\prod_{k!=i}(1-\phi(d(x,x_k)))]}{\partial w} \tag{27}$$

The derivative of the product sequence is calculated as,

$$\frac{\partial(\prod_{k!=i}(1-\phi(d(x_i,x_k))))}{\partial w} = -\sum_{\substack{h=1\\h!=i}}^{n}\frac{\partial(\phi(x_i,x_h))}{\partial w}\cdot\prod_{\substack{k=1\\k!=h}}(1-\phi(x_i,x_k)) \tag{28}$$

Finally, the derivative of the mass assigned to the domain can be calculated as,

$$\frac{\partial m^*(x_i)}{\partial w} = \frac{K\frac{\partial(\prod_{i=1}^{N}(1-\phi(d(x,x_i))))}{\partial w} - \frac{\partial K}{\partial w}\prod_{i=1}^{N}(1-\phi(d(x,x_i)))}{K^2} \tag{29}$$

Then we can apply this calculated gradient, predicting the target variable y_i of every example e_i in the training set $\mathcal{L}$. When we predict the output value of a sample e_i, we leave out this sample from set

$\mathcal{L}$ and use all the other $n-1$ samples for which the actual output value is known in the set as evidence. We will perform this operation multiple times through all the training set, a single pass through all the training set will be called an epoch. The algorithm will perform a fixed number of epochs. At the end of each epoch the weights will be updated by the calculated gradient multiplied by a learning rate factor α and we will store the weight vector only if a local minimum is found. Finally, the method will return the weight vector $\hat{w}$ which minimizes the loss function L, as we show in Algorithm 1.

Algorithm 1 Weight learning.

```
function WEIGHT LEARNING(X, Y, α, NumEpochs)
    w ← [1, 1, ..., 1]
    minLoss ← ∞
    ŵ ← w
    for epoch ← (1, NumEpochs) do
        Ŷ ← [ ]
        for x_i ∈ X do
            S ← Remove(x_i, X)                    ▷ Remove x_i from training set
            ŷ_i ← EVREG(x_i, S, w)                ▷ Predict example e_i with training set S
            Insert(ŷ_i, Ŷ)                        ▷ Insert ŷ_i in Ŷ
        end for
        loss ← L(Y, Ŷ)                            ▷ Compute loss
        gradient ← CalculateGradient(Y, Ŷ)        ▷ Compute gradient
        w ← w + α * gradient                      ▷ Update weights vector
        if minLoss > loss then                    ▷ Save best weight vector
            minLoss ← loss
            ŵ ← w
        end if
    end for
    return ŵ
end function
```

The introduction of the weight learning process makes the EVREG model not only better predict the output of a given variable, but also gain the ability to perform feature selection tasks. The main advantage of EVREG is its transparency. This means we can easily track any prediction made by the model; we can get the contribution (mass) of each piece of evidence in the training set $\mathcal{L}$, which makes it intrinsically interpretable. The model can now present a ranking of features according to their importance (weight), i.e., the contribution of a feature to each mass computed, thus helping the end user get a better understanding of her/his data.

5. Synthetic Data Experiments

We use synthetic datasets in this section to show the prediction capabilities of WEVREG, testing its prediction performance in various configurations. First we compare its performance with the ones of other well-known regression methods previously used in the health care costs prediction problem. The synthetic datasets that will be used are detailed in Table 3.

Table 3. Synthetic regression datasets.

Name	Samples	Relevant Features	Total Features
Linear Regression	200	5	500
Friedman	200	5	500
Linear Regression 5k	5000	5	500
Friedman 5k	5000	5	500

These datasets were previously used by Bugata and Drotár [41] to test the performance of WkNN. WkNN is a k-NN based algorithm that, like our method, finds the weight of each feature and then uses a k-NN regressor to make a prediction. WkNN will be one of the methods that will be compared to WEVREG.

The Linear Regression dataset is generated using a random linear regression model, then a gaussian noise with deviation 1 is applied to the output. The Friedman regression problem is a synthetic dataset described in [58], which has only 5 relevant features. The input is uniformly distributed on the interval $[0, 1]$. Again a gaussian noise with 1 standard deviation is applied; the formula is shown in Equation (30).

$$f(x) = 10sin(\pi x_0 x_1) + 20(x_2 - 0.5)^2 + 10x_3 + 5x_4 + N(0, 1) \tag{30}$$

We used the Mean Absolute Error (MAE) to objectively measure the performance of each method. This error computes the average absolute difference between the predicted cost $\hat{y}$ and the real value y, as shown in Equation (31). Where $\hat{y}$ and y are vectors of size n.

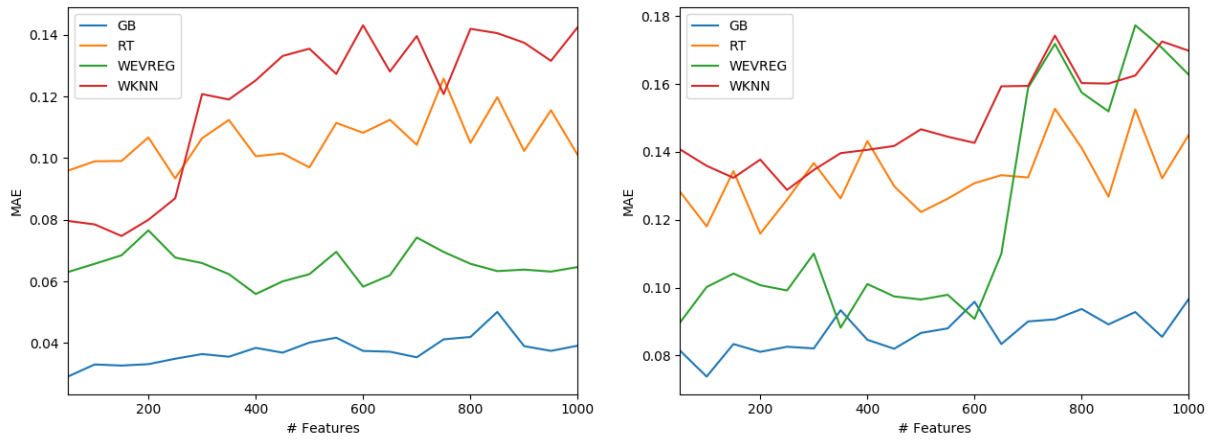
$$MAE(\hat{y}, y) = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}_i - y_i| \tag{31}$$

We used three regression methods to compare their performance to the one of WEVREG. The first method was WkNN due to its similarity with WEVREG. Then the two tree-based algorithms to be used were RT and GB . Every method had a unique configuration in each set for simplicity. We used a grid search approach to find the best parameter for each method, except for WkNN where the same configuration as WEVREG was used to demonstrate the difference between both methods. WEVREG used the closest 20 neighbors with a learning rate of 0.1 iterating for 100 times. RT considered all features to create a split and 1 was the minimum number of samples required to be at a leaf node. For GB we used 500 boosting stages and 4 as the minimum number of samples to split a node, with a learning rate of 0.1. The output was scaled to be bound to $[0, 1]$. The performance of these methods is shown in Table 4.

Table 4. MAE on synthetic datasets (the lower the better).

Name	RT	WkNN	WEVREG	GB
Linear Regression	0.11 ± 0.03	0.08 ± 0.02	0.08 ± 0.02	$\mathbf{0.03} \pm 0.01$
Friedman	0.12 ± 0.01	0.14 ± 0.01	0.10 ± 0.01	$\mathbf{0.08} \pm 0.01$
Linear Regression 5k	0.06 ± 0.01	0.02 ± 0.00	0.03 ± 0.00	$\mathbf{0.01} \pm 0.00$
Friedman 5k	0.09 ± 0.01	0.08 ± 0.01	0.07 ± 0.01	$\mathbf{0.05} \pm 0.01$

GB was the method that obtained the best overall performance as can be observed in Table 4. In particular, WEVREG obtained a good performance resulting in the second best method in most of the sets. WEVREG obtained similar results to WkNN, with slightly better performance of WEVREG. Furthermore, we tested the performance of the methods with a variable number of features. We could observe the performance of each method in the Linear Regression and Friedman dataset using between 50 and 1000 features in Figure 4.



(**a**) Linear Regression. (**b**) Friedman.

Figure 4. Model performance with different number of features.

As shown in Figure 4, WEVREG did not decrease its performance with the inclusion of more features in the Linear Regression problem (Figure 4a). This was not the case in the Friedman dataset, where it had an abrupt drop when the 600 features were exceeded, resulting in a performance similar to WkNN. However, WEVREG was able to maintain its performance with the increase of unimportant features for a good portion of the tests, which was more than necessary in the real problem we were trying to solve.

6. Experiments and Results

We had access to claims sent by Tsuyama Chuo hospital to the National database and patients' yearly health checkups, as mentioned in Section 3. We crossed data from both sources to obtain:

- **Demographics**: Patients' gender and age.
- **Patients' attributes**: General information about patients such as height, weight, body fat and waist measurement.
- **Health checks**: Results from health check exams a patient had undergone. Japanese workers undergo these exams annually by law. Each exam is indexed by a code, and the result is also included. Some examples are creatinine levels and blood pressure. There were 28 different types of exams, and the date when they were collected was also included.
- **Diagnosis**: Diagnosis for a patient illness registered by date and identified by their ICD-10 codes.
- **Medications**: Detailed information of the dosage and medicine administered, including dates and charges.
- **Costs**: Billing information of each patient's treatment. Including medicine, procedures and hospitalization costs. The sum of these costs in a year is the value that was predicted.

The goal of our experiments was to test the model presented here and compare it to the results that most successful models reported by the up-to-date literature in terms of accuracy and ability to interpret the results. In this work we tried to predict the costs of each patient in the future year. Figure 5 shows the distribution of patients' costs. The chart shows that costs had the same distribution as described in [7], with a spike at 0 and a long right-hand tail.

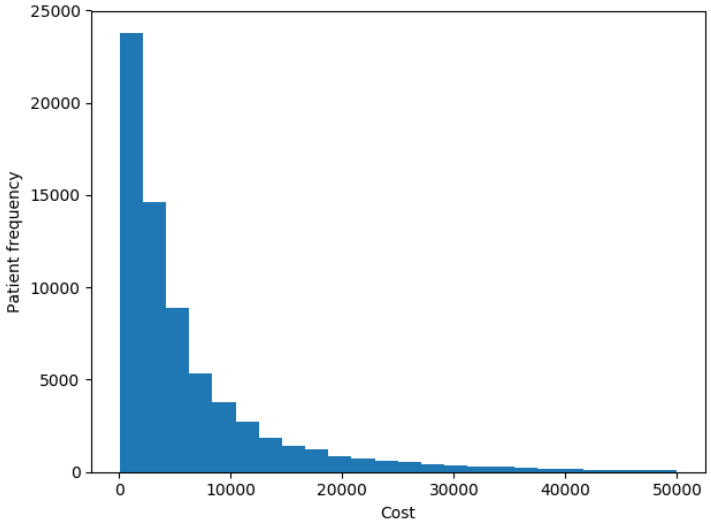


Figure 5. Patients costs distribution.

It has been reported [11–13] that the use of clinical features (health checks, diagnosis and medications) yields the same performance as using only cost features. This conclusion has been reached without the proper use of an FS algorithm, just by testing the use and omission of clinical data, thus it is not clear whether or not specific clinical data helps to determine a patient's costs. We will use the clinical features to properly test this hypothesis.

Encoding a patient's history was done by using all sources available as features. The sources are demographics, health checkups, diagnosis in ICD-10 codification, previous and actual costs.We will only consider chronic conditions for the diagnosis as these can be carried from one year to the next. We will use as chronic conditions the diseases defined by Koller et al. [59] in a study of the impact of multi-morbidity on long-term care dependency. In the study they used 46 chronic conditions based on ICD-10 codes defined by the Central Research Institute of Statutory Ambulatory Health Care in Germany. Table 5 shows a description of the patient's vector representation with the number of variables of each type in each scenario. As an input vector, we used all dimensions shown in Table 5 except for the actual cost that was used as our target variable.

Table 5. Number of variables by type in patient encoding.

Description	Scenario 1	Scenario 2	Scenario 3
Demographics	2	2	2
Health checkups	27	54	135
Chronic diseases	46	92	230
Medication info	2	4	10
Previous costs	1	2	5
Actual cost	1	1	1

The evaluation of the performance of our model was done by comparing its results with the methods reported by Sushmita et al. [12], Morid et al. [7] and Duncan et al. [13] for the health cost prediction problem; these works used RT, GB and ANN methods respectively. We also tested a similar regressor algorithm: WkNN.

We used the MAE (Equation (31)) to measure the performance of each method. However, the MAE is useful to compare algorithms in the same set but not to compare results in different datasets, so we also used the Mean Absolute Percentage Error (MAPE), a modified absolute error where the difference between the predicted variable $\hat{y}$ and the real value y is divided by value y; it is computed as:

$$MAPE(\hat{y}, y) = \frac{1}{n} \sum_{i=1}^{n} \frac{|\hat{y}_i - y_i|}{y_i} \tag{32}$$

where again n is the size of vectors $\hat{y}$ and y. One disadvantage of this error measure is that it does not support zero values in the output of a dataset. It is expected that most individuals did not incur any cost (healthy individuals). This not the case for our dataset, as we have information of individuals that have concurred to Tsuyama Chuo Hospital for treatment or for a health check. Therefore, we can use MAPE as a performance measure.

We also used another measure, R^2 which is the Pearson correlation between the predicted and actual health care cost. It represents the proximity of the predicted cost curve to the real cost curve. This value is calculated as:

$$R^2(\hat{y}, y) = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{m})^2} \tag{33}$$

where $\bar{m}$ is the mean of variable y defined as:

$$\bar{m} = \frac{1}{n} \sum_{i=1}^{n} y_i \tag{34}$$

We transformed our target variable (actual costs) to its logarithmic value in each one of the scenarios, as recommended by Diehr et al. [5], so the distribution of patients' costs in Figure 5 was distributed as shown in Figure 6.

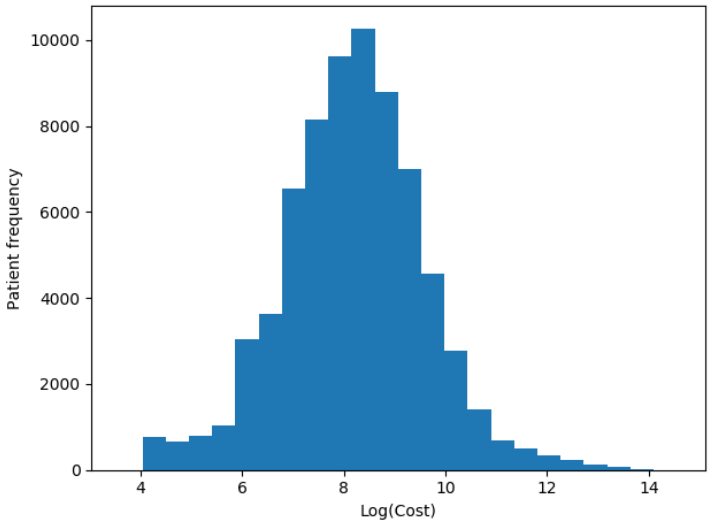


Figure 6. Patients logarithmic costs distribution.

We used a grid search approach for the GB model. We found that the best performance for our dataset was reached when the maximum depth of the individual estimators was 1, the minimum number of samples required to split an internal node was 2 and the number of boosting stages was 125 with a learning

rate of 0.1. We tried multiple configurations (number of layers, number of neurons in each layer and activation function) for the ANN; the best configuration had two hidden layers, the first with double the number of inputs and the other with 10, the learning rate was set to 0.01, and with a Rectified Linear Unit ($f(x) = max(0, x)$) as an activation function. We let it iterate for 30 epochs so it did not overfit. Concerning WkNN configuration, we used 20 neighbors with a Euclidean distance and a RBF kernel. The learning rate was set to 0.01 and was let to iterate for a maximum of 30 epochs. Finally, the WEVREG configuration was the following: we trained the fixed weights for each dimension using gradient descent with a learning rate of 0.1 for 25 epochs, and we used the closest 20 neighbors to improve prediction times, the same quantity used in WkNN.

We performed a 5-fold cross validation procedure to evaluate the performance of each model [60]. The 5-fold cross validation is a statistical procedure where the dataset is randomly divided into five groups, then one of these groups is treated as a validation set and the other four become the training set. This validation is then performed with every group for a total of five times. Finally, the mean performance for each validation group is calculated to obtain the performance of each algorithm. We performed the 5-fold validation five times (resulting in 25 validations) for every tested method. The median of each performance measure is shown in Table 6 with its corresponding standard deviation.

Table 6. Models performance with all features (the lower is the better for Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE); the higher the better for R^2).

Model	Scenario 1			Scenario 2			Scenario 3		
	MAE	**MAPE**	R^2	**MAE**	**MAPE**	R^2	**MAE**	**MAPE**	R^2
RT	0.97 ± 0.01	0.13 ± 0.00	0.16 ± 0.01	0.94 ± 0.01	0.12 ± 0.00	0.18 ± 0.02	0.91 ± 0.02	0.11 ± 0.00	0.17 ± 0.02
WkNN	0.95 ± 0.01	$\mathbf{0.12} \pm 0.00$	0.21 ± 0.01	0.91 ± 0.00	0.12 ± 0.00	0.25 ± 0.00	0.82 ± 0.01	0.10 ± 0.00	0.34 ± 0.01
ANN	0.92 ± 0.01	0.13 ± 0.00	0.22 ± 0.01	0.85 ± 0.01	$\mathbf{0.11} \pm 0.00$	0.30 ± 0.01	0.79 ± 0.03	0.10 ± 0.00	0.35 ± 0.04
WEVREG	0.92 ± 0.02	$\mathbf{0.12} \pm 0.00$	0.23 ± 0.02	$\mathbf{0.84} \pm 0.00$	$\mathbf{0.11} \pm 0.00$	$\mathbf{0.33} \pm 0.00$	0.75 ± 0.02	$\mathbf{0.09} \pm 0.00$	0.41 ± 0.02
GB	$\mathbf{0.89} \pm 0.02$	$\mathbf{0.12} \pm 0.00$	$\mathbf{0.26} \pm 0.02$	$\mathbf{0.84} \pm 0.01$	$\mathbf{0.11} \pm 0.00$	0.32 ± 0.01	$\mathbf{0.67} \pm 0.02$	$\mathbf{0.09} \pm 0.00$	$\mathbf{0.49} \pm 0.02$

Every method increased its prediction performance through every scenario even though the sets shrank in every step; this could be attributed to the availability of longer history for every patient in the sets. RT was the only method that did not increase its performance drastically with each scenario, obtaining the overall worst performance. The best results across all scenarios were obtained by GB. WEVREG obtained similar results to GB, even obtaining a slightly better R^2 in the second scenario, but it had a noticeable difference in the last scenario that could be attributed to the smaller size of the set. WkNN obtained worst metrics compared to WEVREG across all scenarios, although it used the same similarity function and similar weight learning. The higher complexity of the masses computation in WEVREG seemed to benefit its prediction power compared to WkNN.

The most important characteristic of WEVREG was its FS capabilities. Weights for each input feature were learned during its training phase. Each feature started with an initial weight of 1, and it was updated in every iteration of the gradient descent. These weights represented the importance of each feature to predict the target variable which in this case was patients' health care costs. After obtaining the weight of each feature we could select a sub-sample of the features by setting a threshold. We set this threshold to 1, so every weight that increased a little of its value from its starting value would be considered. The top five features on each scenario are shown in Table 7.

Table 7. Top 5 features for each scenario.

Scenario 1		Scenario 2		Scenario 3	
Feature	**Weight**	**Feature**	**Weight**	**Feature**	**Weight**
cost_1	65.06	*cost_1*	19.67	*cost_5*	19.66
age	6.19	*cost_2*	16.47	*cost_4*	19.66
gender	1.09	*age*	3.09	*cost_3*	19.66
dementia_1	1.03	*urinary_incontinence_1*	1.70	*cost_2*	19.66
parkinsons_disease_1	1.03	*dementia_2*	1.59	*diabetes_mellitus_3*	2.43

As we can observe across each scenario, the most noticeable features were cost features. WEVREG assigned larger weights to the closest previous costs. These results are in line with previous works [11–13], where it was reported that cost features alone are a good indicator for future health expenses.

Patient's age seemed to be a small differentiable feature, that had a small effect in the search of similar patients. Diagnosis features and health checkups features did not have a noticeable effect for health care costs prediction. In the case of the last scenario, even though it was overshadowed by cost features, diabetes mellitus was a meaningful diagnosis to reach a correct cost prediction. This could be related to the fact that patients with this diagnosis incurred high expenses attributed to inpatient care [61,62]. An important feature across all scenarios was the diagnosis of dementia in the previous year; it seemed to be a small factor to group patient with similar costs in cases where it was present.

We used the weights learned by the model to filter out unnecessary features. The selected features were the ones that had been assigned a weight larger than 1. Table 8 shows the number of selected features by scenario.

Table 8. Number of features selected by scenario.

Scenario	Total Features	Selected Features
1	76	5
2	154	10
3	382	33

We tested once more the performance of the methods with significantly smaller sets in each scenario. The results can be seen on Table 9.

Table 9. Model performance with selected features (for MAE and MAPE lower is better, for R^2 higher is better).

Model	Scenario 1			Scenario 2			Scenario 3		
	MAE	**MAPE**	R^2	**MAE**	**MAPE**	R^2	**MAE**	**MAPE**	R^2
RT	0.92 ± 0.02	0.13 ± 0.00	0.22 ± 0.03	0.87 ± 0.03	0.12 ± 0.00	0.27 ± 0.03	0.81 ± 0.02	0.10 ± 0.00	0.32 ± 0.02
WkNN	$\mathbf{0.91} \pm 0.01$	$\mathbf{0.12} \pm 0.00$	0.23 ± 0.01	$\mathbf{0.82} \pm 0.01$	$\mathbf{0.11} \pm 0.00$	$\mathbf{0.34} \pm 0.01$	0.76 ± 0.02	0.10 ± 0.00	0.41 ± 0.02
ANN	0.92 ± 0.01	0.13 ± 0.00	0.22 ± 0.01	0.83 ± 0.02	$\mathbf{0.11} \pm 0.00$	0.32 ± 0.02	0.72 ± 0.03	0.09 ± 0.00	0.44 ± 0.03
WEVREG	$\mathbf{0.91} \pm 0.01$	$\mathbf{0.12} \pm 0.00$	0.23 ± 0.01	$\mathbf{0.82} \pm 0.01$	$\mathbf{0.11} \pm 0.00$	$\mathbf{0.34} \pm 0.01$	0.72 ± 0.02	0.09 ± 0.00	0.44 ± 0.02
GB	$\mathbf{0.91} \pm 0.01$	$\mathbf{0.12} \pm 0.00$	$\mathbf{0.24} \pm 0.01$	$\mathbf{0.82} \pm 0.01$	$\mathbf{0.11} \pm 0.00$	0.33 ± 0.01	$\mathbf{0.68} \pm 0.02$	$\mathbf{0.08} \pm 0.00$	$\mathbf{0.48} \pm 0.02$

Every model, except GB, increased its performance in all scenarios, as observed in Table 9. In particular, WkNN and RT had a significant improvement in its MAE and R^2 scores. ANN and WEVREG had a high improvement in the last scenario, which could be attributed to the large number of unnecessary

features in the first place. The initial features seemed to induce too much noise for the models to make precise predictions. On the other hand, GB did not seem to be benefited as the other models with the filtered features since it obtained similar performances, showing the GB resilience in the presence of noise in its features compared to the other tested models. These results show the capability of WEVREG to complete FS tasks, allowing us to decrease the number of features considerably, speeding up training times and model performance. They also show that the most important features for health care costs prediction were cost features; other features such as demographics and some diagnosis such us diabetes mellitus helped to determine a patient's costs but were not as indicative as previous costs.

7. Discussion and Conclusions

We presented a new regression method with the ability to do FS tasks; it has comparable prediction performance as state of the art regression methods. It can easily show the most relevant features for health care cost prediction; in particular, we obtained that cost features for our dataset are the most relevant to determine a patient's future health care costs.

Since WEVREG uses a k-NN approach we can easily keep track of how a prediction is made; this is a desirable ability in the health care domain. We compared its results with the predictions made by four models; two of them (GB and ANN) are the best ones from the eleven models analyzed and reported by Morid et al. [7]. When comparing results we can conclude that our method obtains comparable performances to these methods, proving that it is possible to create and use more transparent models for a regression problem like health care cost prediction, challenging the common belief that complex and black-box like methods are always the solution with the best performance for every problem being presented.

We improved Evidential regression presented by Petit-Renaud and Denœux [10] to be used in the prediction of health care costs. Our results were obtained using data of electronic health records from Tsuyama Chuo Hospital. The results showed that our Weighted Evidential Regression method obtained $R^2 = 0.44$ in the best scenario, which shows that a transparent and interpretable method, could perform as current state of the art supervised learning algorithms such as ANN ($R^2 = 0.44$) and GB ($R^2 = 0.49$).

Even though results are similar or better than previous works, we believe our results are still improvable. One of the approaches to improve performance is classifying patients in cost buckets as recommended by various studies [7,11,13]; this strategy results in better performance but escapes the goal of this work, so we can apply this classification process in future work to obtain a patient's risk class as first step to try to improve the performance of WEVREG. Another approach could be the use of a nearness to death feature as it has been found that costs rise sharply with it [26–28]. It is impossible to know a patient's near death status with our current data. We could include census data to create the new feature, obtaining nearness to death by age group taking into account the change in life expectancy depending on patients' age [63]. We can also try to solve the prediction of health care cost using deep learning methods, but this purpose may become feasible with the availability of a larger dataset. Finally, we also plan to apply this model to other regression problems in the health care domain, such as predicting the hospital length of stay and predicting the days of readmission based on each patient's diagnosis and history, which are two classic prediction problems in this domain.

Author Contributions: B.P.: Method development, experimentation, initial manuscript writing. N.B. (Nelson Baloian): Paper design, supervision. J.A.P.: Manuscript correction and completion. S.P.: Testing consultation. H.S.: Initial data handling. N.B. (Nicolas Bersano): Data analysis. All authors have read and agreed to the published version of the manuscript.

Funding: Partial funding for the research was provided by Allm, Inc., Japan.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. WHO. *Public Spending on Health: A Closer Look at Global Trends*; Technical Report; World Health Organization: Geneva, Switzerland, 2018.
2. Garber, A.M.; Skinner, J. Is American health care uniquely inefficient? *J. Econ. Perspect.* **2008**, *22*, 27–50. [CrossRef] [PubMed]
3. Yoo, I.; Alafaireet, P.; Marinov, M.; Pena-Hernandez, K.; Gopidi, R.; Chang, J.F.; Hua, L. Data mining in healthcare and biomedicine: A survey of the literature. *J. Med. Syst.* **2012**, *36*, 2431–2448. [CrossRef] [PubMed]
4. Bilger, M.; Manning, W.G. Measuring overfitting in nonlinear models: A new method and an application to health expenditures. *Health Econ.* **2015**, *24*, 75–85. [CrossRef] [PubMed]
5. Diehr, P.; Yanez, D.; Ash, A.; Hornbrook, M.; Lin, D. Methods for analyzing health care utilization and costs. *Annu. Rev. Public Health* **1999**, *20*, 125–144. [CrossRef]
6. Kronick, R.; Gilmer, T.; Dreyfus, T.; Ganiats, T. CDPS-Medicare: The Chronic Illness and Disability Payment System Modified to Predict Expenditures for Medicare Beneficiaries. Available online: http://cdps.ucsd.edu/CDPS_Medicare.pdf (accessed on 2 May 2020).
7. Morid, M.A.; Kawamoto, K.; Ault, T.; Dorius, J.; Abdelrahman, S. Supervised Learning Methods for Predicting Healthcare Costs: Systematic Literature Review and Empirical Evaluation. In *AMIA Annual Symposium Proceedings*; American Medical Informatics Association: Bethesda, MD, USA: 2017; Volume 2017, p. 1312.
8. Peñafiel, S.; Baloian, N.; Sanson, H.; Pino, J.A. Applying Dempster–Shafer theory for developing a flexible, accurate and interpretable classifier. *Expert Syst. Appl.* **2020**, *148*, 113262. [CrossRef]
9. Shafer, G.; *A Mathematical Theory of Evidence*; Princeton University Press: Princeton, NJ, USA, 1976; Volume 1.
10. Petit-Renaud, S.; Denœux, T. Nonparametric regression analysis of uncertain and imprecise data using belief functions. *Int. J. Approx. Reason.* **2004**, *35*, 1–28. [CrossRef]
11. Bertsimas, D.; Bjarnadóttir, M.V.; Kane, M.A.; Kryder, J.C.; Pandey, R.; Vempala, S.; Wang, G. Algorithmic prediction of health-care costs. *Oper. Res.* **2008**, *56*, 1382–1392. [CrossRef]
12. Sushmita, S.; Newman, S.; Marquardt, J.; Ram, P.; Prasad, V.; Cock, M.D.; Teredesai, A. Population Cost Prediction on Public Healthcare Datasets. Available online: https://dl.acm.org/doi/abs/10.1145/2750511.2750521 (accessed on 3 May 2020).
13. Duncan, I.; Loginov, M.; Ludkovski, M. Testing alternative regression frameworks for predictive modeling of health care costs. *N. Am. Actuar. J.* **2016**, *20*, 65–87. [CrossRef]
14. Panay, B.; Baloian, N.; Pino, J.A.; Peñafiel, S.; Sanson, H.; Bersano, N. Predicting Health Care Costs Using Evidence Regression. *Proceedings* **2019**, *31*, 74. [CrossRef]
15. Mihaylova, B.; Briggs, A.; O'Hagan, A.; Thompson, S.G. Review of statistical methods for analysing healthcare resources and costs. *Health Econ.* **2011**, *20*, 897–916. [CrossRef]
16. Blough, D.K.; Madden, C.W.; Hornbrook, M.C. Modeling risk using generalized linear models. *J. Health Econ.* **1999**, *18*, 153–171. [CrossRef]
17. Leung, S.F.; Yu, S. On the choice between sample selection and two-part models. *J. Econ.* **1996**, *72*, 197–229. [CrossRef]
18. Marshall, A.H.; Shaw, B.; McClean, S.I. Estimating the costs for a group of geriatric patients using the Coxian phase-type distribution. *Stat. Med.* **2007**, *26*, 2716–2729. [CrossRef] [PubMed]
19. Jones, A.M.; *Models for Health Care*; University of York, Centre for Health Economics: York, UK, 2009.
20. Lee, S.M.; Kang, J.O.; Suh, Y.M. Comparison of hospital charge prediction models for colorectal cancer patients: Neural network vs. decision tree models. *J. Korean Med. Sci.* **2004**, *19*, 677–681. [CrossRef] [PubMed]
21. Frees, E.W.; Jin, X.; Lin, X. Actuarial applications of multivariate two-part regression models. *Ann. Actuar. Sci.* **2013**, *7*, 258–287. [CrossRef]
22. Elith, J.; Leathwick, J.R.; Hastie, T. A working guide to boosted regression trees. *J. Anim. Ecol.* **2008**, *77*, 802–813. [CrossRef]
23. Sutton, C.D. Classification and regression trees, bagging, and boosting. *Handb. Stat.* **2005**, *24*, 303–329.

24. Zurada, J.M. *Introduction to Artificial Neural Systems*; West Publishing Company: St. Paul, MN, USA, 1992; Volume 8.
25. Breiman, L. *Classification and Regression Trees*; Routledge: England, UK, 2017.
26. Tanuseputro, P.; Wodchis, W.P.; Fowler, R.; Walker, P.; Bai, Y.Q.; Bronskill, S.E.; Manuel, D. The health care cost of dying: A population-based retrospective cohort study of the last year of life in Ontario, Canada. *PLoS ONE* **2015**, *10*, e0121759. [CrossRef]
27. Howdon, D.; Rice, N. Health care expenditures, age, proximity to death and morbidity: Implications for an ageing population. *J. Health Econ.* **2018**, *57*, 60–74. [CrossRef]
28. von Wyl, V. Proximity to death and health care expenditure increase revisited: A 15-year panel analysis of elderly persons. *Health Econ. Rev.* **2019**, *9*, 9. [CrossRef]
29. Friedman, J.; Hastie, T.; Tibshirani, R. *The Elements of Statistical Learning*; Springer: Berlin/Heidelber, Germany; New York, NY, USA, 2001; Volume 1.
30. Dash, M.; Liu, H. Feature selection for classification. *Intell. Data Anal.* **1997**, *1*, 131–156. [CrossRef]
31. Yu, L.; Liu, H. Redundancy Based Feature Selection for Microarray Data. Available online: http://www.cs.binghamton.edu/~lyu/publications/Yu-Liu04KDD.pdf (accessed on 4 May 2020).
32. Ruiz, R.; Riquelme, J.C.; Aguilar-Ruiz, J.S. Incremental wrapper-based gene selection from microarray data for cancer classification. *Pattern Recognit.* **2006**, *39*, 2383–2392. [CrossRef]
33. Guyon, I.; Gunn, S.; Nikravesh, M.; Zadeh, L.A. *Feature Extraction: Foundations and Applications*; Springer: Berlin/Heidelber, Germany, 2008; Volume 207.
34. Benesty, J.; Chen, J.; Huang, Y.; Cohen, I. Pearson correlation coefficient. In *Noise Reduction in Speech Processing*; Springer: Berlin/Heidelber, Germany, 2009; pp. 1–4.
35. Goutte, C.; Gaussier, E. A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In *European Conference on Information Retrieval*; Springer: Berlin/Heidelber, Germany, 2005; pp. 345–359.
36. Maldonado, S.; Weber, R. A wrapper method for feature selection using support vector machines. *Inf. Sci.* **2009**, *179*, 2208–2217. [CrossRef]
37. Liaw, A.; Wiener, M. Classification and regression by randomForest. *R News* **2002**, *2*, 18–22.
38. Xu, Z.; Huang, G.; Weinberger, K.Q.; Zheng, A.X. Gradient Boosted Feature Selection. Available online: https://alicezheng.org/papers/gbfs.pdf (accessed on 4 May 2020).
39. Robnik-Šikonja, M.; Kononenko, I. Theoretical and empirical analysis of ReliefF and RReliefF. *Mach. Learn.* **2003**, *53*, 23–69. [CrossRef]
40. Navot, A.; Shpigelman, L.; Tishby, N.; Vaadia, E. Nearest Neighbor Based Feature Selection for Regression and Its Application to Neural Activity. Available online: https://papers.nips.cc/paper/2848-nearest-neighbor-based-feature-selection-for-regression-and-its-application-to-neural-activity.pdf (accessed on 4 May 2020).
41. Bugata, P.; Drotár, P. Weighted nearest neighbors feature selection. *Knowl.-Based Syst.* **2019**, *163*, 749–761. [CrossRef]
42. Shafer, G. Dempster's rule of combination. *Int. J. Approx. Reason.* **2016**, *79*, 26–40. [CrossRef]
43. Niu, G.; Yang, B.S. Dempster–Shafer regression for multi-step-ahead time-series prediction towards data-driven machinery prognosis. *Mech. Syst. Signal Process.* **2009**, *23*, 740–751. [CrossRef]
44. Baraldi, P.; Di Maio, F.; Al-Dahidi, S.; Zio, E.; Mangili, F. Prediction of industrial equipment remaining useful life by fuzzy similarity and belief function theory. *Expert Syst. Appl.* **2017**, *83*, 226–241. [CrossRef]
45. WHO. *International Classification of Functioning, Disability and Health: ICF*; World Health Organization: Geneva, Switzerland, 2001.
46. Matsuda, S.; Fujimori, K. The Claim Database in Japan. *Asian Pac. J. Dis. Manag.* **2014**, *6*, 55–59. [CrossRef]
47. Bolón-Canedo, V.; Sánchez-Maroño, N.; Alonso-Betanzos, A. A review of feature selection methods on synthetic data. *Knowl. Inf. Syst.* **2013**, *34*, 483–519. [CrossRef]
48. Aggarwal, C.C.; Hinneburg, A.; Keim, D.A. On the surprising behavior of distance metrics in high dimensional space. In *International Conference on Database Theory*; Springer: Berlin/Heidelber, Germany, 2001; pp. 420–434.
49. Domingos, P. A few useful things to know about machine learning. *Commun. ACM* **2012**, *55*, 78–87. [CrossRef]

50. Fasshauer, G.E.; Zhang, J.G. On choosing "optimal" shape parameters for RBF approximation. *Numer. Algorithms* **2007**, *45*, 345–368. [CrossRef]
51. Mongillo, M. Choosing basis functions and shape parameters for radial basis function methods. *SIAM Undergrad. Res. Online* **2011**, *4*, 2–6. [CrossRef]
52. Smets, P. What is Dempster-Shafer's model. In *Advances in the Dempster-Shafer Theory of Evidence*; Yager, R., Fedrizzi, M., Kacprzyk, J., Eds.; Wiley: Hoboken, NJ, USA, 1994; pp. 5–34.
53. Johnson, J.; Douze, M.; Jégou, H. Billion-Scale Similarity Search with GPUs. Available online: https://arxiv.org/pdf/1702.08734.pdf (accessed on 6 May 2020).
54. Devroye, L. The uniform convergence of nearest neighbor regression function estimators and their application in optimization. *IEEE Trans. Inf. Theory* **1978**, *24*, 142–151. [CrossRef]
55. Atkeson, C.G.; Moore, A.W.; Schaal, S. Locally weighted learning. In *Lazy Learning*; Springer: Berlin/Heidelber, Germany, 1997; pp. 11–73.
56. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
57. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2016**, arXiv:1609.04747.
58. Friedman, J.H. Multivariate adaptive regression splines. *Ann. Stat.* **1991**, *19*, 1–67 [CrossRef]
59. Koller, D.; Schön, G.; Schäfer, I.; Glaeske, G.; van den Bussche, H.; Hansen, H. Multimorbidity and long-term care dependency—A five-year follow-up. *BMC Geriatr.* **2014**, *14*, 70. [CrossRef]
60. James, G.; Witten, D.; Hastie, T.; Tibshirani, R. *An Introduction to Statistical Learning*; Springer: Berlin/Heidelber, Germany, 2013; Volume 112.
61. Jönsson, B. Revealing the cost of Type II diabetes in Europe. *Diabetologia* **2002**, *45*, S5–S12. [CrossRef]
62. Köster, I.; Von Ferber, L.; Ihle, P.; Schubert, I.; Hauner, H. The cost burden of diabetes mellitus: The evidence from Germany—The CoDiM study. *Diabetologia* **2006**, *49*, 1498–1504. [CrossRef] [PubMed]
63. Stearns, S.C.; Norton, E.C. Time to include time to death? The future of health care expenditure predictions. *Health Econ.* **2004**, *13*, 315–327. [CrossRef] [PubMed]