



# A regression model based on the nearest centroid neighborhood

V. García<sup>1</sup> · J. S. Sánchez<sup>2</sup> · A. I. Marqués<sup>3</sup> · R. Martínez-Peláez<sup>4</sup>

Received: 20 February 2018 / Accepted: 7 April 2018 / Published online: 13 April 2018  
© Springer-Verlag London Ltd., part of Springer Nature 2018

## Abstract

The renowned  $k$ -nearest neighbor decision rule is widely used for classification tasks, where the label of any new sample is estimated based on a similarity criterion defined by an appropriate distance function. It has also been used successfully for regression problems where the purpose is to predict a continuous numeric label. However, some alternative neighborhood definitions, such as the surrounding neighborhood, have considered that the neighbors should fulfill not only the proximity property, but also a spatial location criterion. In this paper, we explore the use of the  $k$ -nearest centroid neighbor rule, which is based on the concept of surrounding neighborhood, for regression problems. Two support vector regression models were executed as reference. Experimentation over a wide collection of real-world data sets and using fifteen odd different values of  $k$  demonstrates that the regression algorithm based on the surrounding neighborhood significantly outperforms the traditional  $k$ -nearest neighborhood method and also a support vector regression model with a RBF kernel.

**Keywords** Nearest neighborhood · Regression analysis · Surrounding neighborhood · Symmetry criterion

## 1 Introduction

The nearest neighbor (NN) rule constitutes one of the most popular nonparametric classification models in pattern recognition and machine learning [7]. The general idea behind this technique is very simple and intuitive: if two examples belong to the same class, they should be close enough to

each other according to a measure of dissimilarity in the  $D$ -dimensional feature space  $\mathbb{R}^D$ . Thus, given a data set of size  $n$ ,  $T = \{(X_1, Y_1), (X_2, Y_2), \dots, (X_n, Y_n)\}$ , where  $X_i \in \mathbb{R}^D$  denotes the  $i$ -th training example and  $Y_i \in \{\omega_1, \omega_2, \dots, \omega_M\}$  is its class label, a new sample  $p$  is assigned to the class of its nearest neighbor in the training set  $T$ . An extension to the NN decision rule is the  $k$ -NN classifier, in which the label to be assigned to  $p$  corresponds to the one with a majority of votes from the  $k$  closest examples in  $T$ .

Apart from other properties common to most nonparametric classification techniques, the  $k$ -NN rule combines its conceptual simplicity and good performance with the fact that its asymptotic or infinite ( $n \rightarrow \infty$ ) error tends to the optimal Bayes error under very weak conditions ( $k \rightarrow \infty$  and  $k/n \rightarrow 0$ ).

In general, the  $k$ -NN model has intensively been applied to classification problems with the aim of predicting or estimating a discrete class label. However, this technique has already been used for regression modeling [3, 6, 17, 22] where the labels to be estimated correspond to continuous values. For instance, Yao and Ruzo [26] proposed a general framework based on the  $k$ -NN algorithm for the prediction of gene function. Dell'Acqua et al. [8] introduced the time-aware multivariate NN regression method to predict traffic flow. Treiber and Kramer [23] analyzed the  $k$ -NN regression method in a multivariate times series model for predicting

✉ J. S. Sánchez  
sanchez@uji.es

V. García  
vicente.jimenez@uacj.mx

A. I. Marqués  
imarques@uji.es

R. Martínez-Peláez  
rmartinezp@delasalle.edu.mx

<sup>1</sup> División Multidisciplinaria en Ciudad Universitaria, Universidad Autónoma de Ciudad Juárez, 32310 Ciudad Juárez, Chihuahua, Mexico

<sup>2</sup> Department of Computer Languages and Systems, Institute of New Imaging Technologies, Universitat Jaume I, 12071 Castelló de la Plana, Spain

<sup>3</sup> Department of Business Administration and Marketing, Universitat Jaume I, 12071 Castelló de la Plana, Spain

<sup>4</sup> Facultad de Tecnologías de la Información, Universidad De La Salle Bajío, León, Mexico

the wind power of turbines. Yang and Zhao [25] developed several generalized algorithms of the  $k$ -NN regression and applied them to a face recognition problem. Hu et al. [14] predicted the capacity of lithium-ion batteries by means of a data-driven method based on  $k$ -NN, which is used to build a nonlinear kernel regression model. Xiao et al. [24] combined NN and logistic regression for the early diagnosis of late-onset neonatal sepsis. Eronen and Klapuri [10] proposed an approach for tempo estimation from musical pieces with  $k$ -NN regression. Leon and Popescu [16] presented an algorithm based on large margin NN regression for predicting students performance using their contributions to several social media tools. Yu and Hong [27] developed an ensemble of NN regression in low-rank multi-view feature space to infer 3D human poses from monocular videos.

Intuitively, neighborhood should be defined in a way that the neighbors of a sample are as close to it as possible and they are located as homogeneously around it as possible. The second condition is a consequence of the first in the asymptotic case, but in some practical cases, the geometrical distribution may become even more important than the actual distances to characterize a sample by means of its neighborhood [19]. As the traditional concept of neighborhood takes care of the first property only, the nearest neighbors may not be placed symmetrically around the sample.

Some alternative neighborhoods have been proposed as a way to overcome the problem just pointed out. These consider both proximity and symmetry so as to define the general concept of surrounding neighborhood [19]: they try to search for neighbors of a sample close enough (in the basic distance sense), but also in terms of their spatial distribution with respect to it. The nearest centroid neighborhood [5] is a well-established representative of the surrounding neighborhood, showing a better behavior than the classical nearest neighborhood on a variety of preprocessing and classification tasks [12, 19, 20, 28].

Taking into account the good performance in classification, the purpose of this paper is to introduce the  $k$ -nearest centroid neighbors ( $k$ -NCN) model for regression and to investigate its efficiency by carrying out a comprehensive empirical analysis over 31 real-life data sets when varying the neighborhood size ( $k$ ).

Henceforth, the paper is organized as follows. Section 2 presents the foundations of the  $k$ -NCN algorithm and defines the regression algorithm proposed in this paper. Section 3 provides the main characteristics of the databases and the setup of the experiments carried out. Section 4 discusses the experimental results. Finally, Sect. 5 remarks the main conclusions and outlines possible avenues for future research.

## 2 Regression models based on neighborhood

In this section, we briefly introduce the basis of the regression models based on  $k$ -NN and  $k$ -NCN.

Let  $T = \{(\mathbf{x}_1, a_1), \dots, (\mathbf{x}_n, a_n)\} \in (\mathbf{x} \times a)^n$  be a data set of  $n$  independent and identically distributed (i.i.d.) random pairs  $(\mathbf{x}_i, a_i)$ , where  $\mathbf{x}_i = [x_{i1}, x_{i2}, \dots, x_{iD}]$  represents an example in a  $D$ -dimensional feature space and  $a_i$  denotes the continuous target value associated to it. The aim of regression is to learn a function  $f : \mathbf{y} \rightarrow a$  to predict the value  $a$  for a query sample  $\mathbf{y} = [y_1, y_2, \dots, y_D]$ .

### 2.1 $k$ -NN regression

The concept of the  $k$ -NN rule for regression can be generalized since the nearest neighbor method assigns a new sample  $\mathbf{y}$  the same target value as the closest example in  $T$ , according to a certain dissimilarity measure (generally, the Euclidean distance). An extension of this procedure is the  $k$ -NN decision rule, in which the algorithm retrieves the  $k$  closest examples in  $T$ .

When  $k = 1$ , the target value assigned to the input sample is the target value indicated by its closest neighbor. For  $k > 1$ , the  $k$ -NN regression model ( $k$ -NNR) estimates the target value  $f(\mathbf{y})$  of a new input sample  $\mathbf{y}$  by averaging the target values of its  $k$ -nearest neighbors [2, 13, 15]:

$$f(\mathbf{y}) = \frac{1}{k} \sum_{i=1}^k a_i \quad (1)$$

where  $a_i$  denotes the target value of the  $i$ -th nearest neighbor.

### 2.2 $k$ -NCN regression

Let  $p$  be a query sample whose  $k$ -nearest centroid neighbors should be found from a set  $X = \{x_1, \dots, x_n\}$ . These  $k$  neighbors are such that (a) they are as near  $p$  as possible, and (b) their centroid is also as close to  $p$  as possible. Both conditions can be satisfied through the iterative procedure given in Algorithm 1.

**Algorithm 1** Nearest centroid neighbors

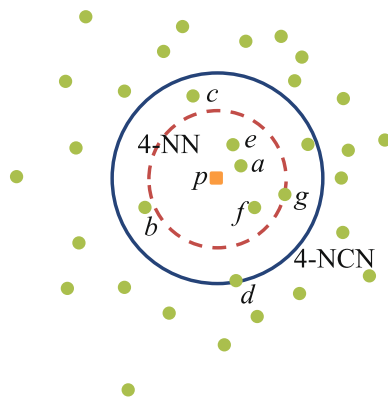
```

1: Input:
2:  $X = \{x_1, \dots, x_n\}$  {Input data set}
3:  $k$  {Neighborhood size}
4:  $p$  {Query point}
5:
6: Output:
7:  $Q = \{q_1, \dots, q_k\}$  {Nearest centroid neighbors}
8:
9:  $Q \leftarrow \emptyset$ 
10:  $q_1 \leftarrow \text{findNN}(X, p)$  { $q_1$  is the nearest neighbor}
11:  $Q \leftarrow \{q_1\}$ 
12:  $Aux \leftarrow X - \{q_1\}$ 
13:
14:  $j \leftarrow 1$ 
15: while  $j < k$  do
16:    $j \leftarrow j + 1$ 
17:    $dist \leftarrow \infty$ 
18:   for all  $x_i \in Aux$  do
19:      $M \leftarrow \text{computeCentroid}(Q \cup \{x_i\})$ 
20:     if  $\text{computeDist}(M, p) < dist$  then
21:        $q_j \leftarrow x_i$ 
22:     end if
23:   end for
24:    $Q \leftarrow Q \cup \{q_j\}$ 
25:    $Aux \leftarrow Aux - \{q_j\}$ 
26: end while

```

The algorithm is better illustrated through a simple example in Fig. 1. The first neighbor of a query point  $p$ , which is denoted by the letter  $a$ , corresponds to its first nearest neighbor. The second neighbor is not the second nearest neighbor (represented as  $e$ ); instead, the algorithm picks a point located in the opposite direction of the first neighbor with respect to  $p$  so that the centroid of that point and all previously selected neighbors is the closest to  $p$ .

This definition leads to a type of neighborhood in which both closeness and spatial distribution of neighbors are taken into account because of the symmetry (centroid) criterion. Besides, the proximity of the nearest centroid neighbors to



**Fig. 1** A comparison between NCN and NN

the sample is guaranteed because of the incremental nature of the way in which those are obtained from the first nearest neighbor. However, note that the iterative procedure outlined in Algorithm 1 does not minimize the distance to the centroid because it gives precedence to the individual distances instead. On the other hand, the region of influence of the NCN results bigger than that of the traditional nearest neighborhood; as can be seen in Fig. 1, the four nearest centroid neighbors ( $a, b, c, d$ ) of a point  $p$  enclose a region quite bigger than the region defined by the four nearest neighbors ( $a, e, f, g$ ).

For a set of cardinality  $n$ , computation of one nearest centroid neighbor of any point requires at most  $n$  centroid and distance computations, and also  $n$  comparisons to find the minimum of the distances. Therefore,  $k$ -nearest centroid neighbors of a point can be computed in  $O(kN)$  time, which is the same as that required for the computation of  $k$ -nearest neighbors.

From the concept of nearest centroid neighborhood, it is possible to introduce an alternative regression model, namely  $k$ -NCNR, which estimates the output of a query sample  $y$  as follows:

1. Find the  $k$ -nearest centroid neighbors of  $y$  by using Algorithm 1.
2. Estimate the target value of  $y$  as the average of the target values of its  $k$  neighbors by means of Eq. 1.

### 3 Experiments

The main purpose of the experiments in this study is twofold. First, we want to establish whether or not the proposed  $k$ -NCNR model outperforms the classical  $k$ -NNR algorithm. Second, we are also interested in evaluating the performance of the best  $k$ -NCNR and  $k$ -NNR algorithms in comparison with two support vector regression methods. Experimentation was carried out over a collection of 31 data sets with a wide variety of characteristics in terms of number of attributes and samples. All these data sets were taken from the KEEL repository [1], and their main characteristics are summarized in Table 1.

The fivefold cross-validation procedure was adopted for the experiments because it provides some advantages over other resampling strategies, such as bootstrap with a high computational cost or re-substitution with a biased behavior [18]. The original data set was randomly divided into five stratified segments or folds of (approximately) equal size; for each fold, four blocks were used to fit the model, and the remaining portion was held out for evaluation as an independent test set. Then the results reported here correspond to the averages across the five trials.

**Table 1** Characteristics of the data sets used in the experiments

		#Samples	#Attributes
(1)	Diabetes	43	2
(2)	Ele-1	495	2
(3)	Plastic	1650	2
(4)	Quake	2178	3
(5)	Laser	993	4
(6)	Ele-2	1056	4
(7)	AutoMPG6	392	5
(8)	Friedman	1200	5
(9)	Delta-Ail	7129	5
(10)	MachCPU	209	6
(11)	Dee	365	6
(12)	AutoMPG8	392	7
(13)	Anacalt	4052	7
(14)	Concrete	1030	8
(15)	Abalone	4177	8
(16)	California	20,640	8
(17)	Stock	950	9
(18)	Wizmir	1461	9
(19)	Wankara	1609	9
(20)	MV	40,768	10
(21)	ForestFire	517	12
(22)	Treasury	1049	15
(23)	Mortgage	1049	15
(24)	Baseball	337	16
(25)	House	22,784	16
(26)	Elevators	16,599	18
(27)	Compact	8192	21
(28)	Pole	14,998	26
(29)	Puma32h	8192	32
(30)	Ailerons	13,750	40
(31)	Tic	9822	85

**Table 2** Parameters of the regression algorithms

Method	Learning parameters
<i>k</i> -NCNR	<i>k</i> = 1, 3, ..., 29; Euclidean distance
<i>k</i> -NNR	<i>k</i> = 1, 3, ..., 29; Euclidean distance
SVR(L1)	Complexity parameter = 1; linear kernel (polynomial of degree 1); sequential minimal optimization algorithm; epsilon round-off error = $1 \times 10^{12}$ ; epsilon insensitive loss function = 0.001; tolerance = 0.001
SVR(RBF)	Complexity parameter = 1; RBF kernel; sequential minimal optimization algorithm; gamma = 0.01; epsilon round-off error = $1 \times 10^{12}$ ; epsilon insensitive loss function = 0.001; tolerance = 0.001

The main hyper-parameters of the regression models used in the experiments are listed in Table 2. Note that two support vector regression (SVR) algorithms [21], with linear and RBF kernels, were also employed as reference solutions for comparison purposes.

### 3.1 Evaluation criteria

In the framework of regression, the purpose of most performance evaluation scores is to estimate how much the predictions  $(p_1, p_2, \dots, p_n)$  deviate from the target values  $(a_1, a_2, \dots, a_n)$ . These metrics are minimized when the predicted value for each query sample agrees with its true value [4]. Probably, the most popular measure that has extensively been used to evaluate the performance of a regression model is the root mean square error (RMSE),

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (p_i - a_i)^2} \tag{2}$$

This metric indicates how far the predicted values  $p_i$  are from the target values  $a_i$  by averaging the magnitude of individual errors without taking care of their sign.

From the RMSE, we defined the error normalized difference, which is computed for each data set  $i$  and each neighborhood size  $k$  as follows:

$$Diff_{error_{i,k}} = \frac{RMSE_{NN_{i,k}} - RMSE_{NCN_{i,k}}}{RMSE_{NN_{i,k}}} \tag{3}$$

where  $RMSE_{NN_{i,k}}$  and  $RMSE_{NCN_{i,k}}$  represent the RMSE achieved on data set  $i$  using  $k$ -NNR and  $k$ -NCNR, respectively.

In practice,  $Diff_{error_{i,k}}$  can be considered as an indicator of improvement or deterioration of the  $k$ -NCNR method with respect to the  $k$ -NNR model:

- if  $Diff_{error_{i,k}} > 0$ ,  $k$ -NCNR is better than  $k$ -NNR;
- if  $Diff_{error_{i,k}} < 0$ ,  $k$ -NCNR is worse than  $k$ -NNR;
- if  $Diff_{error_{i,k}} \approx 0$ , there are no significant differences between  $k$ -NNR and  $k$ -NCNR.

### 3.2 Nonparametric statistical tests

When comparing the results of two or more models over multiple data sets, a nonparametric statistical test is more appropriate than a parametric one because the former is not based on any assumption such as normality or homogeneity of variance [9, 11].

Both pairwise and multiple comparisons were used in this paper. First, we applied the Friedman’s test to discover any statistically significant differences among all the regression models. This starts by ranking the

algorithms for each data set independently according to the RMSE results: as there are 30 competing models (15  $k$ -NNR and 15  $k$ -NCNR), the ranks for each data set are from 1 (best) to 30 (worst). Then the average rank of each algorithm across all data sets is computed.

As the Friedman’s test only detects significant differences over the whole pool of comparisons, we then proceeded with the Holm’s post hoc test in order to compare a control algorithm (the best model) against the remaining techniques by defining a collection of hypothesis around the control method.

Afterward, the Wilcoxon’s paired signed-rank test was employed to find out whether or not there exist significant differences between each pair of the five top  $k$ -NNR and  $k$ -NCNR algorithms. This statistic ranks the differences in performance of two algorithms for each data set, ignoring the signs, and compares the ranks for the positive and the negative differences.

In summary, the statistical tests were used as follows: (i) the Friedman’s test was employed over all the models; (ii) the Wilcoxon’s, Friedman’s and Holm’s post hoc tests were applied to the five top-ranked  $k$ -NNR and  $k$ -NCNR algorithms with the aim of concentrating the analysis on the best results of each approach.

### 4 Results

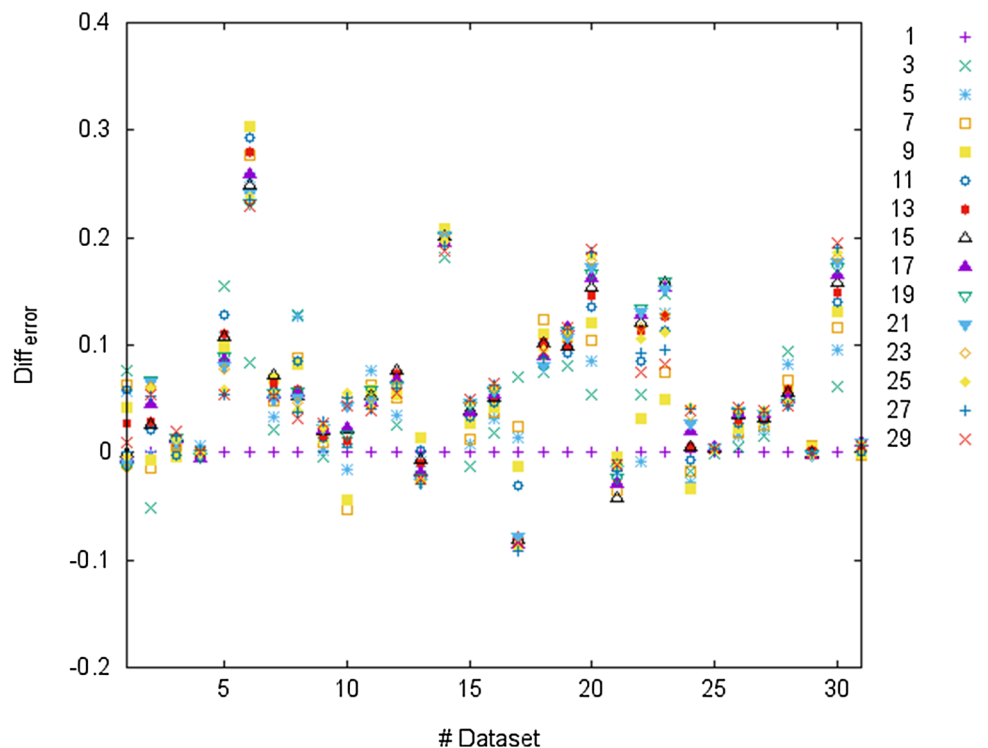
This section is divided into two blocks. First, the comparison between the  $k$ -NCNR and  $k$ -NNR models is discussed in Sect. 4.1. Second, the results of the best configurations of  $k$ -NCNR and  $k$ -NNR are compared against the results of the SVR models in Sect. 4.2. The detailed results obtained over each data set and each algorithm are reported in Tables 8 and 9 in the Appendix.

#### 4.1 $k$ -NCNR versus $k$ -NNR

Figure 2 depicts the error normalized difference for each database ( $i = 1, \dots, 31$ ) with all neighborhood sizes. The most important observation is that a vast majority of cases achieved positive values ( $Diff_{error,k} > 0$ ), indicating that the performance of the  $k$ -NCNR model was superior to that of the corresponding  $k$ -NNR algorithm for most databases.

Figure 3 shows the Friedman’s average ranks achieved from the RMSE results with all the regression methods ( $k$ -NNR and  $k$ -NCNR). As can be observed, the lowest (best) average ranks were achieved with both strategies using  $k$  values in the range from 9 to 21. More specifically, the best  $k$ -NNR configurations were with  $k = 9, 11, 17, 13, 15$ , whose ranks were 6.4194, 6.7097, 6.7903, 6.9032 and 6.9032, respectively. In the case of  $k$ -NCNR, the best  $k$  values were 11, 19, 21, 9 and 13 with ranks 7.0323, 7.1935, 7.2581, 7.3226 and 7.3871, respectively.

Fig. 2 Error normalized difference on the 31 data sets



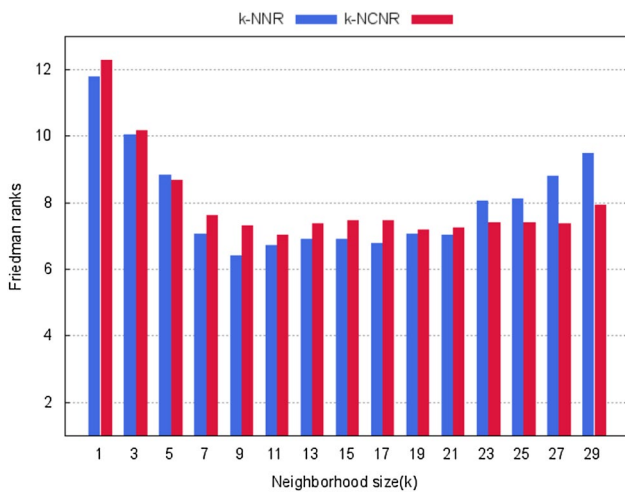


Fig. 3 Friedman’s ranks of the *k*-NNR and *k*-NCNR models

Table 3 reports the results of the Wilcoxon’s test applied to the ten best regression models. The upper diagonal half summarizes this statistic at a significance level of  $\alpha = 0.10$  (10% or less chance), and the lower diagonal half corresponds to a significance level of  $\alpha = 0.05$ . The symbol “•” indicates that the method in the row significantly outperforms the method in the column, whereas the symbol “o” means that the method in the column performs significantly better than the method in the row.

Analysis of the results in Table 3 allows to remark that the *k*-NCNR models were significantly better than the *k*-NNR algorithms. On the other hand, it is also interesting to note that different values of *k* did not yield statistically significant differences between pairs of the same strategy; for instance, in the case of *k*-NCNR, there was no neighborhood size performing significantly better than some other value of *k*.

Because the Wilcoxon’s test for multiple comparisons does not allow to conclude which algorithm is the best, we applied a Friedman’s test to the five top-ranked *k*-NNR and *k*-NCNR approaches and afterward, a Holm’s post

hoc test in order to determine whether or not there exists significant differences with the best (control) model. As we had 10 algorithms and 31 databases, the Friedman’s test using the Iman–Davenport statistic, which is distributed according to the *F*-distribution with  $10 - 1 = 9$  and  $(10 - 3)(31 - 1) = 270$  degrees of freedom, was 8.425717. The *p* value calculated by  $F(9, 270)$  was  $4.3 \times 10^{-11}$  and therefore, the null hypothesis that all algorithms performed equally well can be rejected with a high significance level.

Figure 4 depicts the Friedman’s average rankings for the five top-ranked *k*-NNR and *k*-NCNR algorithms. One can see that the approach with the best scores corresponds to 11-NCNR, which will be the control algorithm for the subsequent Holm’s post hoc test. It is also worth pointing out that all the *k*-NCNR models achieved lower rankings than the *k*-NNR methods, proving the superiority of the surrounding neighborhood to the conventional neighborhood,

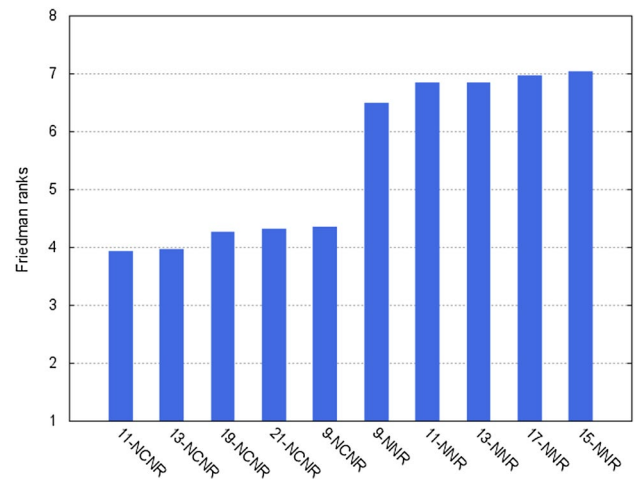


Fig. 4 Friedman’s ranks of the five best results for the *k*-NNR and *k*-NCNR models

Table 3 Summary of the Wilcoxon’s statistic for the best *k*-NNR and *k*-NCNR models

		(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
(1)	9-NNR	–					o			o	o
(2)	11-NNR		–				o	o	o	o	o
(3)	17-NNR			–			o	o	o	o	o
(4)	13-NNR				–		o	o	o	o	o
(5)	15-NNR					–	o	o	o	o	o
(6)	11-NCNR	•	•	•	•	•	–				
(7)	19-NCNR		•	•	•	•		–			
(8)	21-NCNR		•	•	•	•			–		
(9)	9-NCNR	•	•	•	•	•				–	
(10)	13-NCNR	•	•	•	•	•					–

Upper and lower diagonal halves are for  $\alpha = 0.10$  and  $\alpha = 0.05$ , respectively

**Table 4** Unadjusted  $p$  values for  $\alpha = 0.05$  and  $\alpha = 0.10$  with 11-NCNR as the control algorithm

	$z$	$p$ value	$\alpha = 0.05/i$	$\alpha = 0.10/i$
<b>15-NNR</b>	4.026889	0.000057	0.005556	0.011111
<b>17-NNR</b>	3.942996	0.000080	0.006250	0.012500
<b>21-NNR</b>	3.775209	0.000160	0.007143	0.014286
<b>13-NNR</b>	3.775209	0.000160	0.008333	0.016667
<b>9-NNR</b>	3.313794	0.000920	0.010000	0.020000
9-NCNR	0.545308	0.585542	0.012500	0.025000
21-NCNR	0.503361	0.614710	0.016667	0.033333
19-NCNR	0.419468	0.674874	0.025000	0.050000
13-NCNR	0.041947	0.966541	0.050000	0.100000

The models in bold were significantly worse than the control algorithm

Table 4 reports the results of the Holm’s test using 11-NCNR as the control algorithm, including the  $z$  value, the unadjusted  $p$  value, and the adjusted  $\alpha$  value at significance levels of 0.05 and 0.10. It can be viewed that 11-NCNR was significantly better than the five top-ranked  $k$ -NNR models at both significance levels. On the contrary, it is not possible to reject the null hypothesis of equivalence between 11-NCNR and the rest of  $k$ -NCNR algorithms.

### 4.2 Neighborhood-based regression models versus SVR

This section analyzes the results of the two top  $k$ -NCNR and  $k$ -NNR algorithms with respect to two SVR algorithms. The average RMSE results of these models on the 31 data sets and the Friedman’s average rankings are reported in Table 5.

Friedman’s average ranks for the four regressions models are plotted in Fig. 5. As can be seen, both 11-NCNR and SVR(L1) arose as the algorithms with the lowest rankings, that is, the lowest RMSE in average.

In order to check whether or not the RMSE results were significantly different, the Iman–Davenport’s statistic was computed. This is distributed according to an  $F$ -distribution with 3 and 90 degrees of freedom. The  $p$  value computed was 0.03275984862, which is less than a significance level of  $\alpha = 0.05$ . Therefore, the null hypothesis that all regression models performed equally well can be rejected.

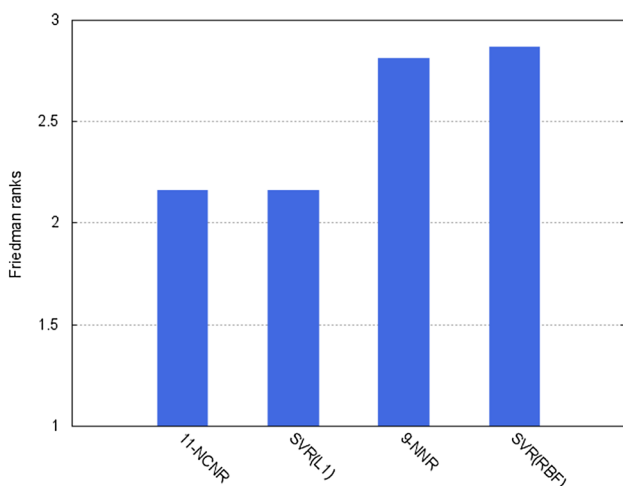
Table 6 shows the unadjusted  $p$  values for a Holm’s post hoc test using the 11-NCNR algorithm as the control method. For a significance level of  $\alpha = 0.05$ , the procedure could not reject the null hypothesis of equivalence in any of the three algorithms. Conversely, at a significance level of  $\alpha = 0.10$ , the Holm’s test indicates that 11-NCNR was significantly better than 9-NNR and SVR(RBF), and equivalent to SVR(L1).

**Table 5** Average RMSE results of two SVR models and the best neighborhood-based algorithms

	SVR(L1)	SVR(RBF)	11-NCNR	9-NNR
Diabetes	$5.88 \times 10^{-1}$	$6.62 \times 10^{-1}$	$6.16 \times 10^{-1}$	$6.31 \times 10^{-1}$
Ele-1	$6.40 \times 10^2$	$7.64 \times 10^2$	$6.41 \times 10^2$	$6.41 \times 10^2$
Plastic	$1.53 \times 10^0$	$2.19 \times 10^0$	$1.59 \times 10^0$	$1.63 \times 10^0$
Quake	$2.04 \times 10^{-1}$	$2.04 \times 10^{-1}$	$1.94 \times 10^{-1}$	$1.95 \times 10^{-1}$
Laser	$2.34 \times 10^1$	$2.46 \times 10^1$	$1.06 \times 10^1$	$1.14 \times 10^1$
Ele-2	$1.68 \times 10^2$	$2.16 \times 10^2$	$1.19 \times 10^2$	$1.60 \times 10^2$
AutoMPG6	$3.55 \times 10^0$	$3.67 \times 10^0$	$3.88 \times 10^0$	$4.14 \times 10^0$
Friedman	$2.71 \times 10^0$	$2.69 \times 10^0$	$1.68 \times 10^0$	$1.81 \times 10^0$
Delta-Ail	$1.74 \times 10^{-4}$	$1.76 \times 10^{-4}$	$1.85 \times 10^{-4}$	$1.90 \times 10^{-4}$
MachCPU	$6.98 \times 10^1$	$9.03 \times 10^1$	$8.04 \times 10^1$	$7.53 \times 10^1$
Dee	$4.08 \times 10^{-1}$	$4.23 \times 10^{-1}$	$4.02 \times 10^{-1}$	$4.22 \times 10^{-1}$
AutoMPG8	$3.45 \times 10^0$	$3.61 \times 10^0$	$3.92 \times 10^0$	$4.20 \times 10^0$
Anacalt	$5.15 \times 10^{-1}$	$5.14 \times 10^{-1}$	$7.67 \times 10^{-2}$	$7.80 \times 10^{-2}$
Concrete	$1.11 \times 10^1$	$1.09 \times 10^1$	$7.91 \times 10^0$	$9.64 \times 10^0$
Abalone	$2.27 \times 10^0$	$2.40 \times 10^0$	$2.12 \times 10^0$	$2.20 \times 10^0$
California	$7.08 \times 10^4$	$7.33 \times 10^4$	$9.17 \times 10^4$	$9.67 \times 10^4$
Stock	$2.39 \times 10^0$	$2.48 \times 10^0$	$9.23 \times 10^{-1}$	$8.46 \times 10^{-1}$
Wizmir	$1.26 \times 10^0$	$1.27 \times 10^0$	$1.31 \times 10^0$	$1.45 \times 10^0$
Wankara	$1.57 \times 10^0$	$1.58 \times 10^0$	$1.36 \times 10^0$	$1.48 \times 10^0$
MV	$5.31 \times 10^0$	$2.35 \times 10^0$	$6.08 \times 10^0$	$7.07 \times 10^0$
ForestFire	$5.71 \times 10^1$	$5.71 \times 10^1$	$5.87 \times 10^1$	$5.97 \times 10^1$
Treasury	$2.48 \times 10^{-1}$	$2.85 \times 10^{-1}$	$5.44 \times 10^{-1}$	$5.17 \times 10^{-1}$
Mortgage	$5.31 \times 10^0$	$2.35 \times 10^0$	$3.72 \times 10^{-1}$	$3.54 \times 10^{-1}$
Baseball	$7.57 \times 10^2$	$7.76 \times 10^2$	$9.08 \times 10^2$	$8.92 \times 10^2$
House	$4.77 \times 10^4$	$4.79 \times 10^4$	$5.02 \times 10^4$	$5.07 \times 10^4$
Elevators	$2.97 \times 10^{-3}$	$2.93 \times 10^{-3}$	$6.35 \times 10^{-3}$	$6.56 \times 10^{-3}$
Compact	$1.24 \times 10^1$	$1.35 \times 10^1$	$6.23 \times 10^0$	$6.50 \times 10^0$
Pole	$3.10 \times 10^1$	$3.28 \times 10^1$	$8.20 \times 10^0$	$8.33 \times 10^0$
Puma32H	$2.71 \times 10^{-2}$	$2.70 \times 10^{-2}$	$2.79 \times 10^{-2}$	$2.82 \times 10^{-2}$
Ailerons	$1.77 \times 10^{-4}$	$1.70 \times 10^{-4}$	$3.00 \times 10^{-4}$	$3.49 \times 10^{-4}$
Tic	$2.44 \times 10^{-1}$	$2.44 \times 10^{-1}$	$2.39 \times 10^{-1}$	$2.41 \times 10^{-1}$
Diabetes	$5.88 \times 10^{-1}$	$6.62 \times 10^{-1}$	$6.16 \times 10^{-1}$	$6.31 \times 10^{-1}$
Ele-1	$6.40 \times 10^2$	$7.64 \times 10^2$	$6.41 \times 10^2$	$6.41 \times 10^2$
Plastic	$1.53 \times 10^0$	$2.19 \times 10^0$	$1.59 \times 10^0$	$1.63 \times 10^0$
Quake	$2.04 \times 10^{-1}$	$2.04 \times 10^{-1}$	$1.94 \times 10^{-1}$	$1.95 \times 10^{-1}$
Laser	$2.34 \times 10^1$	$2.46 \times 10^1$	$1.06 \times 10^1$	$1.14 \times 10^1$
Ele-2	$1.68 \times 10^2$	$2.16 \times 10^2$	$1.19 \times 10^2$	$1.60 \times 10^2$
AutoMPG6	$3.55 \times 10^0$	$3.67 \times 10^0$	$3.88 \times 10^0$	$4.14 \times 10^0$
Friedman	$2.71 \times 10^0$	$2.69 \times 10^0$	$1.68 \times 10^0$	$1.81 \times 10^0$
Delta-Ail	$1.74 \times 10^{-4}$	$1.76 \times 10^{-4}$	$1.85 \times 10^{-4}$	$1.90 \times 10^{-4}$
MachCPU	$6.98 \times 10^1$	$9.03 \times 10^1$	$8.04 \times 10^1$	$7.53 \times 10^1$
Dee	$4.08 \times 10^{-1}$	$4.23 \times 10^{-1}$	$4.02 \times 10^{-1}$	$4.22 \times 10^{-1}$
AutoMPG8	$3.45 \times 10^0$	$3.61 \times 10^0$	$3.92 \times 10^0$	$4.20 \times 10^0$
Anacalt	$5.15 \times 10^{-1}$	$5.14 \times 10^{-1}$	$7.67 \times 10^{-2}$	$7.80 \times 10^{-2}$
Concrete	$1.11 \times 10^1$	$1.09 \times 10^1$	$7.91 \times 10^0$	$9.64 \times 10^0$
Abalone	$2.27 \times 10^0$	$2.40 \times 10^0$	$2.12 \times 10^0$	$2.20 \times 10^0$
California	$7.08 \times 10^4$	$7.33 \times 10^4$	$9.17 \times 10^4$	$9.67 \times 10^4$

**Table 5** (continued)

	SVR(L1)	SVR(RBF)	11-NCNR	9-NNR
Stock	$2.39 \times 10^0$	$2.48 \times 10^0$	$9.23 \times 10^{-1}$	$8.46 \times 10^{-1}$
Wizmir	$1.26 \times 10^0$	$1.27 \times 10^0$	$1.31 \times 10^0$	$1.45 \times 10^0$
Wankara	$1.57 \times 10^0$	$1.58 \times 10^0$	$1.36 \times 10^0$	$1.48 \times 10^0$
MV	$5.31 \times 10^0$	$2.35 \times 10^0$	$6.08 \times 10^0$	$7.07 \times 10^0$
ForestFire	$5.71 \times 10^1$	$5.71 \times 10^1$	$5.87 \times 10^1$	$5.97 \times 10^1$
Treasury	$2.48 \times 10^{-1}$	$2.85 \times 10^{-1}$	$5.44 \times 10^{-1}$	$5.17 \times 10^{-1}$
Mortgage	$5.31 \times 10^0$	$2.35 \times 10^0$	$3.72 \times 10^{-1}$	$3.54 \times 10^{-1}$
Baseball	$7.57 \times 10^2$	$7.76 \times 10^2$	$9.08 \times 10^2$	$8.92 \times 10^2$
House	$4.77 \times 10^4$	$4.79 \times 10^4$	$5.02 \times 10^4$	$5.07 \times 10^4$
Elevators	$2.97 \times 10^{-3}$	$2.93 \times 10^{-3}$	$6.35 \times 10^{-3}$	$6.56 \times 10^{-3}$
Compact	$1.24 \times 10^1$	$1.35 \times 10^1$	$6.23 \times 10^0$	$6.50 \times 10^0$
Pole	$3.10 \times 10^1$	$3.28 \times 10^1$	$8.20 \times 10^0$	$8.33 \times 10^0$
Puma32H	$2.71 \times 10^{-2}$	$2.70 \times 10^{-2}$	$2.79 \times 10^{-2}$	$2.82 \times 10^{-2}$
Ailerons	$1.77 \times 10^{-4}$	$1.70 \times 10^{-4}$	$3.00 \times 10^{-4}$	$3.49 \times 10^{-4}$
Tic	$2.44 \times 10^{-1}$	$2.44 \times 10^{-1}$	$2.39 \times 10^{-1}$	$2.41 \times 10^{-1}$
Avg. Ran.	2.16	2.87	2.16	2.81



**Fig. 5** Friedman’s ranks of two best  $k$ -NCNR and  $k$ -NNR benchmarked methods and two SVR models

**Table 6** Unadjusted  $p$  values for  $\alpha = 0.05$  and  $\alpha = 0.10$  with 11-NCNR as the control algorithm when compared against 9-NNR, SVR(L1), and SVR(RBF)

	$z$	$p$ value	$\alpha = 0.05/i$	$\alpha = 0.10/i$
<b>SVR (RBF)</b>	2.164225	0.030447	0.016667	0.033333
9-NNR	1.967478	0.049128	0.025	0.05
SVR (L1)	0	1	0.05	0.10

The model in bold was significantly worse than the control algorithm at  $\alpha = 0.10$

**Table 7** Summary of the Wilcoxon’s statistic for the best  $k$ -NNR and  $k$ -NCNR models, and two SVR algorithms

		(1)	(2)	(3)	(4)
(1)	11-NCNR	–	•		•
(2)	9-NNR	◦	–		
(3)	SVR (L1)			–	•
(4)	SVR (RBF)			◦	–

Upper and lower diagonal halves are for  $\alpha = 0.10$  and  $\alpha = 0.05$ , respectively

We run a Wilcoxon’s paired signed-rank test for  $\alpha = 0.05$  and  $\alpha = 0.10$  between each pair of regression algorithms. From Table 7, we can observe that 11-NCNR performed significantly better than 9-NNR at both significance levels, and it was significantly better than SVR(RBF) at  $\alpha = 0.10$ . On the other hand, it also has to be noted that SVR(L1) was significantly better than the SVR model with an RBF kernel at  $\alpha = 0.10$  and  $\alpha = 0.05$ . This suggests that, for regression problems, we can use either  $k$ -NCNR or the linear SVR, since both these models yielded equivalent performance results.

### 5 Conclusions and future work

In this paper, a new regression technique based on the nearest centroid neighborhood has been introduced. The general idea behind this strategy is that neighbors of a query sample should fulfill two complementary conditions: proximity and symmetry. In order to discover the applicability of this regression model, it has been compared to the  $k$ -NNR algorithm when varying the neighborhood size  $k$  from 1 to 29 (using only the odd values) and two configurations of SVR (with linear and RBF kernels) over a total of 31 databases.

The experimental results in terms of RMSE (and the error normalized difference proposed here) have shown that the  $k$ -NCNR model is statistically better than the  $k$ -NNR method. In particular, the best results have been achieved with values of  $k$  in the range from 9 to 21 and more specifically, the 11-NCNR approach has outperformed the five top-ranked  $k$ -NNR algorithms. When compared against the two SVR models, the results have suggested that the  $k$ -NCNR algorithm performs equally well as the linear SVR and better than SVR(RBF).

It is also important to note that the  $k$ -NCNR model is a lazy algorithm that does not require any training, which can constitute an interesting advantage over the SVR methods for big data applications.

Several promising directions for further research have emerged from this study. First, a natural extension is to develop regression models based on other surrounding







neighborhoods such as those defined from the Gabriel graph and the relative neighborhood graph, which are two well-known proximity graphs. Second, it would be interesting to assess the performance of the  $k$ -NCNR algorithm and compared to other regression models when applied to some real-life problem.

**Acknowledgements** This work has partially been supported by the Generalitat Valenciana under Grant [PROMETEOII/2014/062] and the Spanish Ministry of Economy, Industry and Competitiveness under Grant [TIN2013-46522-P].

## Appendix

Tables 8 and 9 report the average RMSE results for all the data sets and for each value of  $k$ . In addition, the Friedman's rankings are given in the last row of each table.

## References

- Alcalá-Fdez J, Fernández A, Luengo J, Derrac J, García S, Sánchez L, Herrera F (2011) KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *J Mult-Valued Log Soft Comput* 17:255–287
- Biau G, Devroye L, Dujmović V, Krzyżak A (2012) An affine invariant  $k$ -nearest neighbor regression estimate. *J Multivar Anal* 112:24–34
- Buza K, Nanopoulos A, Nagy G (2015) Nearest neighbor regression in the presence of bad hubs. *Knowl-Based Syst* 86:250–260
- Caruana R, Niculescu-Mizil A (2004) Data mining in metric space: an empirical analysis of supervised learning performance criteria. In: 10th ACM SIGKDD international conference on knowledge discovery and data mining. New York, pp 69–78
- Chaudhuri B (1996) A new definition of neighborhood of a point in multi-dimensional space. *Pattern Recognit Lett* 17(1):11–17
- Cheng CB, Lee E (1999) Nonparametric fuzzy regression  $k$ -nn and kernel smoothing techniques. *Comput Math Appl* 38(3–4):239–251
- Dasarathy B (1990) Nearest neighbor (NN) norms: NN pattern classification techniques. IEEE Computer Society Press, Los Alamitos
- Dell'Acqua P, Bellotti F, Berta R, Gloria AD (2015) Time-aware multivariate nearest neighbor regression methods for traffic flow prediction. *IEEE Trans Intell Transp Syst* 16(6):3393–3402
- Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res* 7:1–30
- Eronen AJ, Klapuri AP (2010) Music tempo estimation with  $k$ -nn regression. *IEEE Trans Audio Speech Lang Process* 18(1):50–57
- García S, Fernández A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. *Inf Sci* 130:2044–2064
- García V, Sánchez JS, Martín-Félez R, Mollineda RA (2012) Surrounding neighborhood-based SMOTE for learning from imbalanced data sets. *Prog Artif Intell* 1(4):347–362
- Guyader A, Hengartner N (2013) On the mutual nearest neighbors estimate in regression. *J Mach Learn Res* 14:2361–2376
- Hu C, Jain G, Zhang P, Schmidt C, Gomadam P, Gorka T (2014) Data-driven method based on particle swarm optimization and  $k$ -nearest neighbor regression for estimating capacity of lithium-ion battery. *Appl Energy* 129:49–55
- Lee SY, Kang P, Cho S (2014) Probabilistic local reconstruction for  $k$ -nn regression and its application to virtual metrology in semiconductor manufacturing. *Neurocomputing* 131:427–439
- Leon F, Popescu E (2017) Using large margin nearest neighbor regression algorithm to predict student grades based on social media traces. In: International conference in methodologies and intelligent systems for technology enhanced learning. Porto, Portugal, pp 12–19
- Mack YP (1981) Local properties of  $k$ -nn regression estimates. *SIAM J Algebr Discrete Methods* 2(3):311–323
- Ounpraseuth S, Lensing SY, Spencer HJ, Kodell RL (2012) Estimating misclassification error: a closer look at cross-validation based methods. *BMC Res Notes* 5(656):1–12
- Sánchez JS, Marqués AI (2002) Enhanced neighbourhood specifications for pattern classification. In: Pattern recognition and string matching, pp 673–702
- Sánchez JS, Pla F, Ferri FJ (1998) Improving the  $k$ -NCN classification rule through heuristic modifications. *Pattern Recognit Lett* 19(13):1165–1170
- Shevade SK, Keerthi SS, Bhattacharyya C, Murthy KRK (2000) Improvements to the SMO algorithm for SVM regression. *IEEE Trans Neural Netw* 11(5):1188–1193
- Song Y, Liang J, Lu J, Zhao X (2017) An efficient instance selection algorithm for  $k$  nearest neighbor regression. *Neurocomputing* 251(16):26–34
- Treiber N, Kramer O (2015) Evolutionary feature weighting for wind power prediction with nearest neighbor regression. In: IEEE congress on evolutionary computation. Sendai, Japan, pp 332–337
- Xiao Y, Griffin MP, Lake DE, Moorman JR (2010) Nearest-neighbor and logistic regression analyses of clinical and heart rate characteristics in the early diagnosis of neonatal sepsis. *Med Decis Making* 30(2):258–266
- Yang S, Zhao C (2006) Regression nearest neighbor in face recognition. In: 18th International conference on pattern recognition. Hong Kong, China, pp 515–518
- Yao Z, Ruzo W (2006) A regression-based  $k$  nearest neighbor algorithm for gene function prediction from heterogeneous data. *BMC Bioinform* 7(1):1–11
- Yu J, Hong C (2017) Exemplar-based 3D human pose estimation with sparse spectral embedding. *Neurocomputing* 269:82–89
- Zhang J, Yim YS, Yang J (1997) Intelligent selection of instances for prediction functions in lazy learning algorithms. *Artif Intell Rev* 11(1–5):175–191