Stratification by Quantiles and Compact Genetic Algorithms for Dataset Compression in Nonlinear Regression

Estratificación por Cuantiles y Algoritmos Genéticos Compactos para la Compresión de Conjuntos de Datos en Regresión No Lineal

Uriel Calderon-Uribe[1], Salvador Calderon-Uribe[2], Victor Ayala-Ramirez[1], Jose-Gabriel Aguilera-Gonzalez[1], Antonio Tierrasnegras-Badillo[1]

[1]División de Sistemas Computacionales, Tecnológico Nacional de México/ITS Sur de Guanajuato, Av. Educación Superior No. 2000, Col. Juárez, Uriangato 38982, Guanajuato, México. u.calderon@itsur.edu.mx

[2]Laboratorio de Visión Artificial y Termografía/Mecatrónica Facultad de Ingeniería, Universidad Autónoma de Querétaro, San Juan del Rio, México.

Abstract

K-nearest neighbors (KNN) is a supervised learning technique that predicts the outcome of a sample based on the samples surrounding it. One of the main drawbacks of this method is the memory required for prediction. All samples must be stored, so the inference time increases with the size of the dataset. For this reason, this work presents a technique based on genetic algorithms (GA) to select the most relevant samples, thereby reducing the complexity of the KNN algorithm. The main objective of GA is to use the samples that contribute most to the prediction in a regression system, reducing the number of samples and the computational load of KNN. The results of this work show a significant improvement in inference times when combining GA with the KNN method.

Palabras clave: K-Nearest Neighbors, Genetic Algorithms, Supervised Classification, Instance based-learning

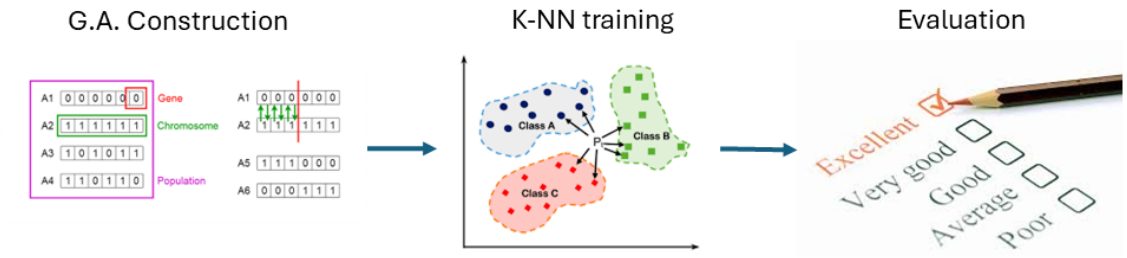
1. Introduction

K-nearest neighbors is a supervised learning method where; to infer a new sample, the surrounding neighbors play an important role in the prediction (Ahmed *et al.,* 2020). For regression, the prediction method consists of using the K nearby samples and, through a statistical process (generally calculating the mean or median), obtaining the new value. However, one of the disadvantages of this model is that, to calculate a new instance, all samples in the dataset must be queried. This results in large amounts of memory being required to store the information (Valero *et al.,* 2017). These problems have led to various studies focusing on selecting the instances that best represent the dataset. Techniques based on bio-inspired algorithms have demonstrated superiority in choosing the best features from a dataset, showing that robustness can be improved without compromising classification metrics (Cano *et al.,* 2005). Other methods involve the use of clustering, which attempts to obtain the best representation of the dataset based on its density (Garcia *et al.,* 2012). Although these techniques allow for obtaining representative samples of the dataset, finding a balance between sample size and the value of K remains challenging. For this reason, this study uses a system that drastically reduces both the number of instances and the value of K, while preserving classification indices and the robustness of KNN. This method is based on the construction of a genetic algorithm whose main feature is a modification to the chromosome structure (Garcia *et al.,* 2008). The chromosome now has the peculiarity of being a vector containing samples of a particular class. With these additional features, a KNN-based regressor is constructed where the value of K is restricted to three. The results of these modifications position this method as a tool that can be implemented in an embedded system.

2. Proposed Methodology

This section describes the methods used to construct the KNN regression system in conjunction with the GA. Figure 1 illustrates the steps used to build the final model. As shown in this figure, the system is divided into three steps: GA construction, KNN training, and evaluation. In the first step, GA construction, a subset of the dataset is used by the GA to find the samples that best describe the regression system. Once these

samples are found, a KNN model is trained on them, minimizing the value of K. Finally, this model is evaluated using a test set to measure its efficiency.



2.1. Genetic Algorithm design

The main goal of GA is to minimize the size of the dataset without degrading the performance of the regression system. To achieve this, the total training values of y were divided into B disjoint blocks using percentiles, ensuring that each block initially contained the same information density:

$$B_1, B_2, \dots, B_B \quad (1)$$

Each block B_B is delimited by a lower and upper threshold based on the actual signal distribution. Thus, the optimization problem for the regression system is defined as follows:

$$\min MSE(P) : P \subset D_d \;\; y\,|P_b| = p_b, \forall b = 1, \dots, B \quad (2)$$

Where $MSE(P)$ is the mean squared error, P_b is the subset of selected prototypes, and p_b is the fixed budget of samples allowed for block b. To carry out the mutation process, for each gene within block B_B, the method restricts the search for the new sample only to the indices of the original dataset whose values of y belong to that same statistical block. On the other hand, the crossover operator was modified so that the cutoff points preferentially coincide with the block boundaries ($p_b, 2p_b, 3p_b, \dots$). This preserves the behavioral banks that a parent has already learned to solve well. Finally, the fitness function seeks to minimize the error between the prediction and the actual value. For this, the fitness function is defined according to (equation 3) :

$$Fitness\ (P) = \frac{1}{|D_t|} \sum_{(x,y)\in D_t} (KNN_{reg}(x|P)\ -\ y\ )^2$$

Where $KNN_{reg}\ (x\,|\,P)$ is the continuous prediction by neighborhood average returned by the model trained with the strict prototype budget P.

2.2. KNN model construction

In machine learning, KNN is a supervised learning model that bases its inference method on the proximity of the data to the sample being predicted. Thus, in a regression system, to predict a new sample, KNN uses Euclidean distance to determine how far each of the training samples is from the reduced set of prototypes. Therefore, Euclidean distance is defined according to equation 4:

$$d(x^*, x_i)\ = \sqrt{\textstyle\sum_{j=1}^{d}(x_j^* - x_{i,j})^2}\ \ (4)$$

Where x^* is the data vector with features ($x^*\ = [x_1, x_2, \dots, x_d]$), and $x_{i,j}$ represents the prediction result. Once all distances have been calculated, they are ordered from smallest to largest. The set of K nearest neighbor indices is defined as $N_k(x^*)$, which satisfies the following condition.

$$d(x^*, x_i) \le d(x^*, x_m) \forall\ x_i \in\ N_k(x^*)\ (5)$$

The size of this set is exactly $|N_k(x^*)| = k$. Finally, for the prediction of a new value, the arithmetic mean of the target values (y_i) of the k selected neighbors are used (equation 6):

$$y^* = \frac{1}{k} \sum_{x_i \in N_{k(x^*)}} y^i \quad (6)$$

3. Resultados

To evaluate the effectiveness of the proposed system, several tests were conducted. First, a linear function was proposed, which served as a starting point for measuring the model's adaptability. Subsequently, a sinusoidal function was introduced to measure the model's behavior in a nonlinear environment. Finally, a real-world problem, based on predicting housing costs, was used to measure the system's robustness.

3.1 Inferential analysis

To carry out the inferential process, a comparative method was proposed. First, a baseline model (a KNN model that uses all the data for training) and the proposed model are used to measure their final performance. Figure 2 shows the results obtained by comparing the baseline model with the proposed model.

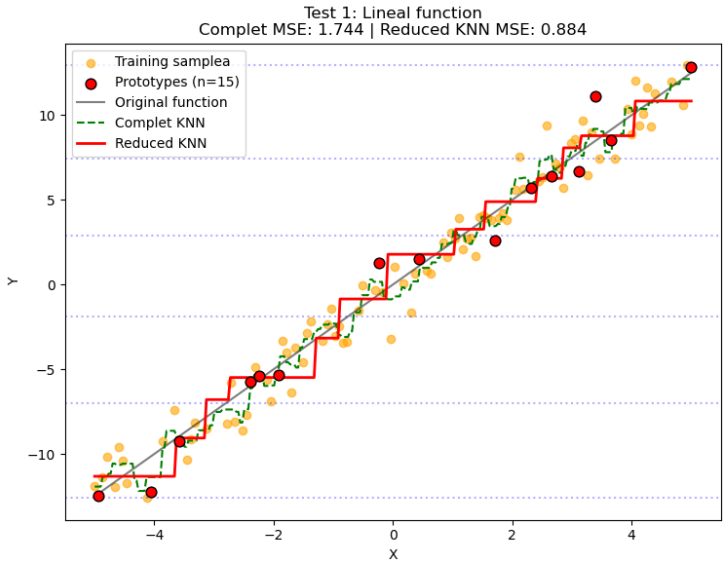


Figure2. Results obtained by comparing the base model with the proposed KNN model.

As can be seen in this figure, both models perform well. The main difference lies in the fact that the proposed model uses only 5 samples for each subdivision of the linear function, for a total of 25 samples to describe the linear function. On the other hand, the base KNN model uses all 1000 samples to predict the behavior of the linear function.

The second experiment involved observing the behavior of the proposed model after modeling a nonlinear system. In this case, the nonlinear function to be modeled is a sinusoidal function, due to its unique behavior over time. Figure 3 shows a comparison between the proposed model and the base model when modeling the sinusoidal function.

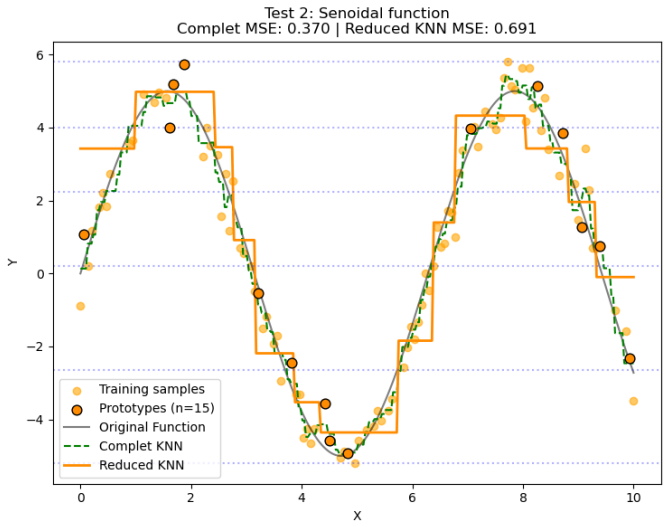


Figure 3. Comparative results when modeling the sinusoidal function with the base model and the proposed model.

Finally, the proposed model was tested on a real-world problem involving predicting home prices in California. This dataset consists of 11 features, and the ultimate goal is to predict house prices based on this information. Figure 4 shows the results obtained using the baseline model compared to the proposed model.

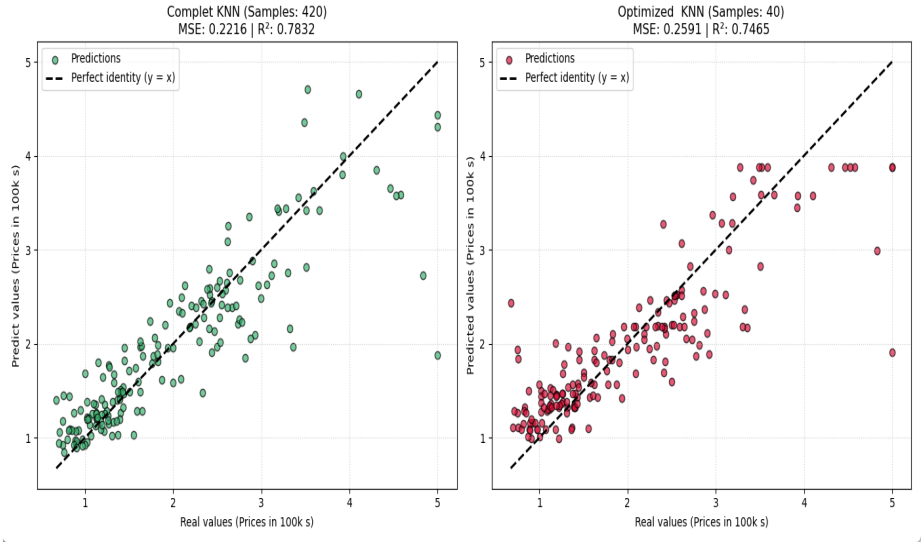


Figure 4. Results obtained in the dataset California housing price (Chen et al. 2024).

As can be seen, the proposed model performs similarly to the base model, clearly demonstrating its superior computational performance. Finally, Table 1 shows the mean squared error (MSE) values for all the test cases described above.

Table 1: Comparison of the MSE through the proposed model and the base model

Model	KNN base model MSE	Reduced KNN MSE
Linear function	1.744	0.8844
No lineal function (sinusoidal)	0.370	0.691
California housing prices dataset	0.2216	0.2591

As can be seen in this table, the proposed model is slightly superior in the analysis of nonlinear functions. However, the proposed model performs very well even with the limited amount of available information.

4. Conclusion

In this study, a combination of GA and KNN was explored to reduce the computational load generated by KNN. The method uses GA to select a reduced set of instances to minimize the mean squared error in a regression system. The results of this study show a similar outcome to calculating the MSE using a standard KNN method and the proposed model. However, the amount of physical space required for each sample and the inference speed were significantly improved. Future work will focus on applying this technique to embedded systems to maximize the expected results while minimizing computational resources.

References

Ahmed, M., Seraj, R., & Islam, S. M. S. (2020). The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics,* 9(8), 1295. https://doi.org/10.3390/electronics9081295

Cano, J. R., Herrera, F., & Lozano, M. (2005). Instance selection using evolutionary algorithms: an experimental study. In Advanced Techniques in Knowledge Discovery and Data Mining (pp. 127-152). London: Springer London. https://doi.org/10.1007/1-84628-183-0_5

Chen, A. (2024). Deep Learning in Real Estate Prediction: An Empirical Study on California House Prices. The National High School Journal of Science. https://nhsjs.com/2024/deep-learning-in-real-estate-prediction-an-empirical-study-on-california-house-prices/

Garcia, S., Derrac, J., Cano, J., & Herrera, F. (2012). Prototype selection for nearest neighbor classification: Taxonomy and empirical study. IEEE transactions on pattern analysis and machine intelligence, 34(3), 417-435. 10.1109/TPAMI.2011.142

García, S., Cano, J. R., & Herrera, F. (2008). A memetic algorithm for evolutionary prototype selection: A scaling up approach. Pattern Recognition, 41(8), 2693-2709. https://simidat.ujaen.es/es/bibcite/reference/245

Olvera López, J. A., Carrasco Ochoa, J. A., & Martínez Trinidad, J. F. (2010). Prototype selection methods. *Computación y Sistemas,* 13(4), 449-462. http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-55462010000200008&lng=es&tlng=en.

Valero-Mas, J. J., Calvo-Zaragoza, J., Rico-Juan, J. R., & Iñesta, J. M. (2017). A study of prototype selection algorithms for nearest neighbour in class-imbalanced problems. In Iberian Conference on Pattern Recognition and Image Analysis (pp. 335-343). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-58838-4_37