A New Weighted k-Nearest Neighbor Algorithm based on Newton's Gravitational Force

Juan Aguilera[1], Luis C. González[1],
Manuel Montes-y-Gómez[2], and Paolo Rosso[3]

[1] Universidad Autónoma de Chihuahua, Chihuahua, Mexico
{p271672,lcgonzalez}@uach.mx
[2] Instituto Nacional de Astrofísica, Óptica y Electrónica, Puebla, Mexico
mmontesg@inaoep.mx
[3] Universitat Politècnica de València, Valencia, Spain
prosso@dsic.upv.es

Abstract. The kNN algorithm has three main advantages that make it appealing to the community: it is easy to understand, it regularly offers competitive performance and its structure can be easily tuning to adapting to the needs of researchers to achieve better results. One of the variations is weighting the instances based on their distance. In this paper we propose a weighting based on the Newton's gravitational force, so that a mass (or relevance) has to be assigned to each instance. We evaluated this idea in the kNN context over 13 benchmark data sets used for binary and multi-class classification experiments. Results in F_1 score, statistically validated, suggest that our proposal outperforms the original version of kNN and is statistically competitive with the distance weighted kNN version as well.

1 Introduction

The k-Nearest Neighbor (kNN) classification algorithm is one of the most popular approaches used by researchers and practitioners in the areas of Pattern Recognition and Machine Learning. Altogether with the Support Vector Machine (SVM), it is considered a firm representative of the classification *by analogy* principle [4].

Generally speaking, kNN only needs one parameter to be adjusted, k, which represents how many closest neighbors are to be considered to classify an unseen object. Once this parameter is set, two main approaches are followed in order to classify an object, (i), the vote of the majority of the k neighbors, and (ii), a weighted vote of all k neighbors considering the distance from where each of them are located with respect to the object to classify. Following these two ideas, the kNN algorithm has been successfully applied in such diverse learning task such as data mining [14], image processing [6], and recommender systems [7].

For classification purposes, all kNN variants, up to now, have assumed that, independently of the voting strategy that they follow (by majority or weighted) all objects in the training set are equal in their classification power. For instance, if two objects from different classes are exactly at the same distance of a test object, both objects will contribute the same amount to the final decision. Another way to perceive this is by saying that the two training objects have the same relevance. In this work, we are interested in proposing some ideas to alter this behavior. Motivated by how big bodies exert and influence to proximate objects, we think of assigning a *mass* to each of the objects in the training set.

There are several scenario applications that make us hypothesize that assigning a mass to all the training objects could have positive effects in the classification performance of the kNN algorithm. Particularly, this could be of interest when some aspect or natural feature of the problem needs to be considered. For example, within the field of Natural Language Processing (NLP), for the task of news classification, capturing the temporal aspect may be relevant, i.e. more recent news could be more informative (or have more context) than older ones[4]. In this case, we could think of the more recent news to have a larger influence, thus a larger mass. Another application of this approach could be the recognition of highly heterogeneous categories. In this case it is usual that the majority of the neighbors (to the object to classify) vote for a wrong label. With objects with different masses it would be possible to overcome this decision, i.e. if the objects with the right class have proper mass.

In this work we approach these ideas by proposing two different ways to calculate a mass for a given object. We formulate the kNN algorithm to take into consideration this mass by using a voting strategy based on Newton's gravitational force. We tested our proposal in 13 benchmark data sets and contrasted the results against the regular kNN and weighted kNN algorithms.

2 Related Work

Literature has reported several ways in which the kNN algorithm could improve its performance. Naturally, finding an optimal value of k has been one of the questions that some works have attempted to solve [17, 16]. Besides finding this k value, there is an open question regarding which distance metric is the more suitable to use. In this regard, some previous works have evaluated new and traditional metrics in a variety of classification problems [2, 15, 8].

Using a weighting scheme was firstly proposed by Dudani [5] in the 70's, this variant of kNN is called the *Distance-Weighted k-Nearest-Neighbor Rule* (DWkNN). Since then, different weighting schemes have been proposed. Among the most recent works, Tan [12] proposed the algorithm *Neighbor-Weighted k-Nearest Neighbor* (NWkNN), which applies a weighting strategy based on the distribution of classes. When working with unbalanced data sets, NWkNN gives a minor weight to objects of majority classes and more weight to objects less represented. For the case of text classification, Soucy and Mineau [11] proposed a weighting based on the similarity of texts (objects), measured by the cosine

[4] Before 2016 it would not be surprising to classify a news containing the term *Donald Trump* in the Business section, when now it would be more appropriate to assign it to the political section.

similarity between their bag-of-word representations. Mateos-García et al. [9] developed a technique similar to those used in Artificial Neural Networks to optimize some weights that would indicate the importance that each neighbor has with respect to the test objects. Finally, Parvinnia et al. [10] also computed a weight for each training object based on a matching strategy between the training and testing data sets.

3 Proposed algorithm

In this section we present two approaches to calculate a mass for a given object in the training set. We then explain the complete kNN framework that exploits the concept of mass, by considering Newton's gravitational force.

3.1 Mass Assignment

Approach 1. Circled by its own class (CC). This approach is based on a instance selection strategy known as Edited Nearest Neighbor (ENN) originally proposed by Wilson [13]. The rationale of ENN is to keep an instance that is surrounded (or circled) by other instances of its same class. For the CC approach, the mass of an object x is directly proportional to the number of objects from its same class that circled it. By doing this, we aim to give less importance to objects that are in regions of the feature space that are more likely to represent a different class. In other words, the idea is to penalize rare objects and, as a consequence, make the classifier more robust to outliers. To calculate the mass via CC we apply the Eq. 1.

$$m(x \in c_i) = \log_2(SN_k(x, c_i) + 2) \tag{1}$$

where x is a training object, c_i is its class and the function $SN_k()$ calculates how many out of the k closest objects to x belong to its same class. The $log_2()$ function serves as a smoothing factor; we include a constant 2 to avoid computation errors or obtaining masses equal to zero.

Approach 2. Circled by different classes (CD). This approach is the opposite of the CC approach. It gives more mass to objects that are surrounded by objects from different classes, that is, the mass is inversely proportional to the number of objects of the same class. CD aims to balance the discriminative power of an outlier object, since it could be relevant to classify other outlier object in the testing set. It also allows to better modeling heterogeneous classes formed by different small subgroups of objects. To assign a mass following this approach we applied the Eq. 2. The interpretation of its elements is the same as in Eq. 1.

$$m(x \in c_i) = \log_2(k - SN_k(x, c_i) + 2) \tag{2}$$

3.2 Weighted Attraction Force kNN algorithm (WAF-kNN)

The traditional weighted kNN algorithm is as follows: given a set of training objects $\{(x_1, f(x_1)), ..., (x_i, f(x_i))\}$ (being x_i an object and $f(x_i)$ its label), an unlabeled object x_q, and the set of the k closest neighbors to x_q in the training set $\{x_1, ..., x_k\}$, the class of x_q is determined by Eq. 3:

$$f(x_q) \leftarrow \arg\max_{c \in C} \sum_{i=1}^{k} weight(x_i) \times \delta(c, f(x_i)) \tag{3}$$

where C represents the set of classes, $weight(x_i)$ indicates the weight for the vote from object x_i, and $\delta(c, f(x_i))$ is a function that returns 1 if x_i belongs to class c or 0, otherwise.

Supported on this framework, our proposal, that we call *Weighted Attraction Force kNN*, or simply WAF-kNN, uses a weighting scheme based on the Law of Universal Gravitation as presented by Eq. 4.

$$weight(x_i) = G\frac{m(x_q)m(x_i)}{dist^2(x_q, x_i)} \simeq \frac{m(x_i)}{dist^2(x_q, x_i)} \tag{4}$$

where $weight(x_i)$ is the attraction force or the voting amount exerted by the training object x_i to classify the object x_q. $m(x_q)$ and $m(x_i)$ are the masses of the testing and training objects respectively, and $dist(\cdot, \cdot)$ is a distance metric between the two objects. The reader could detect that there are two constants that we could omit to simplify the original equation, since they only serve as scaling factors without affecting how the vote is computed. These two constants are G and $m(x_q)$. Note that $m(x_i)$ could be calculated by any of the two approaches, CC or CD, that we already presented in Section 3.1 for mass assignment.

4 Experiments and Results

4.1 Experimental Configuration

For the evaluation of the proposed approach we considered 13 different data sets from the UCI data repository[5]. All these data sets exclusively contain numeric features and do not show any missing value. These data sets are commonly used in classification tasks. Table 1 presents some statistics on these data sets such as the number of instances, features, and classes.

We applied a common experimental setting for the experiments across all the collections. Firstly, we considered three different values for k, namely, 3, 5 and 7. Then, we standardized the data by means of their z-scores. In all the experiments we used the Euclidean distance as the distance measure, and employed the F_1 score as main evaluation metric due to its appropriateness for describing results in unbalanced data sets. A 10-fold cross-validation procedure was applied to get the results. Finally, we applied the non-parametric Bayesian Signed-Rank (BSR) test [1] for analyzing the statistical significance of the obtained results.

[5] https://archive.ics.uci.edu/ml/datasets.html

Table 1. Data sets characteristics.

Data sets	Instances	Features	Classes	Classes Distribution
Arcene	100	10000	2	56/44
Ecoli	336	7	8	143/77/52/35/20/5/2/2
Glass	214	9	6	76/70/29/17/13/9
Haberman	306	3	2	225/81
Ionosphere	351	34	2	225/126
Iris	150	4	3	50/50/50
Landsat	6435	36	6	1533/1508/1358/707/703/626
Page Blocks	5473	10	5	4913/329/115/88/28
Pima	768	8	2	500/268
Sonar	208	60	2	111/97
Thyroid	215	5	3	150/35/30
Vehicle	846	18	4	218/217/212/199
Wine	178	13	3	71/59/48

4.2 Results

Table 2 presents a first comparison of the approaches used to calculate the masses (CC and CD), each employed within the WAF-kNN algorithm. This table is organized by the three k values that were evaluated. The best results, for each k, are shown in bold face. Globally, the CD approach slightly outperforms the CC approach, being this more evident when $k = 7$; notwithstanding, there are data sets where the CC approach is better for all k values, e.g. Arcene and Ecoli. The analysis of the Ecoli data set tell us that classes are more or less well defined in homogeneous clusters. Being this the case, the CD approach gives more mass to *outliers*, causing a larger classification error than CC, which assigns less mass to objects away from their class main centroid and having the effect of reducing noise. Both approaches, CC and CD, aim to offer a better weighting scheme to improve classification performance, but which one to use will ultimately depend on the distribution of classes in the data set of interest.

To evaluate our proposal against kNN and DWkNN algorithms, we chose the CD approach given its consistent performance in the previous experiment. This new comparison is presented in Table 3, where it can be observed that our proposal outperforms the baseline methods in the majority of data sets. This behavior is consistent among the three values of k that are considered. Again, the best performance is obtained with $k = 7$.

To further analyze these results, we applied the non-parametric BSR test [3]. According to this test three possibilities do exist for a given pairwise comparison of methods A and B: (scenario 1) A outperforms B, (scenario 2) both methods show the same performance, or (scenario 3) B outperforms A. The BSR test computes the probability of occurrence of each scenario when we applied approaches A and B over a given data set. Table 4 presents the probabilities of occurrence for each scenario when comparing the baseline approaches kNN and DWkNN with our proposed WAF-kNN approach, respectively.

Table 2. F_1 scores of WAF-kNN, using the two approaches for mass assignment.

Data sets	**CC**	**CD**	**CC**	**CD**	**CC**	**CD**
	k=3		k=5		k=7	
Arcene	**0.762**	0.753	**0.774**	0.758	**0.761**	0.759
Ecoli	**0.714**	0.652	**0.736**	0.706	**0.752**	0.725
Glass	0.556	**0.618**	0.560	**0.613**	0.577	**0.594**
Haberman	**0.571**	0.570	0.535	**0.549**	0.507	**0.528**
Ionosphere	0.796	**0.851**	0.793	**0.826**	0.785	**0.831**
Iris	0.954	0.954	0.954	0.954	0.954	0.954
Landsat	**0.894**	0.883	0.894	**0.895**	0.889	**0.893**
Page Blocks	**0.827**	0.808	**0.826**	0.814	0.815	**0.817**
Pima	**0.697**	0.683	**0.702**	0.692	0.694	**0.695**
Sonar	0.847	**0.863**	0.817	**0.839**	0.803	**0.839**
Thyroid	0.904	**0.933**	0.909	**0.933**	0.909	**0.916**
Vehicle	0.694	**0.707**	0.689	**0.714**	0.688	**0.720**
Wine	0.951	**0.956**	0.969	0.969	0.964	**0.969**

Table 3. Comparison of kNN, DWkNN and WAF-kNN using CD masses.

Data sets	**kNN**	**DWkNN**	**WAF**	**kNN**	**DWkNN**	**WAF**	**kNN**	**DWkNN**	**WAF**
		k = 3			k = 5			k = 7	
Arcene	**0.762**	**0.762**	0.753	**0.796**	**0.796**	0.758	0.736	0.736	**0.759**
Ecoli	0.688	**0.697**	0.652	0.727	**0.729**	0.706	**0.748**	0.747	0.725
Glass	0.610	0.610	**0.618**	0.597	0.604	**0.613**	0.536	0.572	**0.594**
Haberman	0.547	0.561	**0.570**	0.521	0.526	**0.549**	0.524	0.519	**0.528**
Ionosphere	0.797	0.797	**0.851**	0.811	0.813	**0.826**	0.777	0.777	**0.831**
Iris	0.954	0.954	0.954	0.946	**0.954**	**0.954**	0.947	**0.968**	0.954
Landsat	**0.894**	**0.894**	0.883	0.893	0.892	**0.895**	0.889	0.890	**0.893**
Page Blocks	**0.816**	0.814	0.808	0.820	**0.827**	0.814	0.787	**0.817**	0.817
Pima	**0.706**	0.703	0.683	**0.706**	0.704	0.692	0.704	**0.707**	0.695
Sonar	0.847	0.847	**0.863**	0.794	0.798	**0.839**	0.812	0.817	**0.839**
Thyroid	0.904	0.904	**0.933**	0.906	0.909	**0.933**	0.877	0.915	**0.916**
Vehicle	0.706	0.703	**0.707**	0.713	0.711	**0.714**	0.711	0.707	**0.720**
Wine	0.951	0.951	**0.956**	0.964	0.964	**0.969**	0.964	0.964	**0.969**

Table 4. BSR output probabilities. **A** refers to the baseline methods, kNN and DWkNN respectively, whereas **B** refers to the proposed WAF-kNN approach.

Compared algorithms	**Scenarios**		
	A >B	**A = B**	**A <B**
kNN vs WAF-kNN	0.0001	0.1951	0.8048
DWkNN vs WAF-kNN	0.0018	0.6305	0.3677

According to the performance of the WAF algorithm in each data set (with $k = 7$), it was in Ionosphere and Ecoli, where we obtained the largest improvement and decrement with respect to the baseline methods, respectively. When visualizing these data sets, it is possible to notice some data characteristics that could shed some light on details about the behavior of the method.

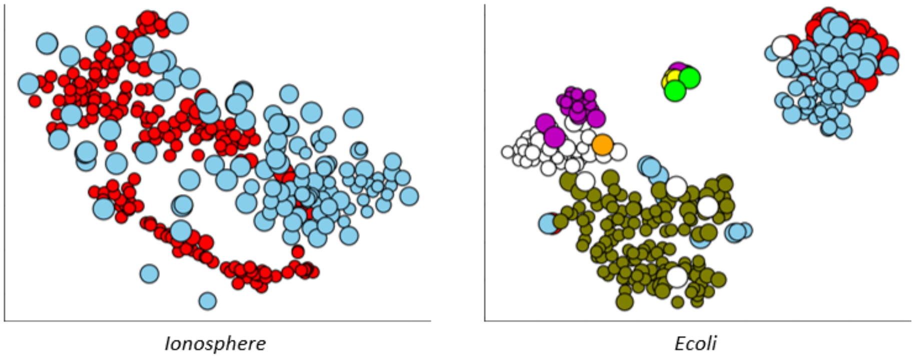

Fig. 1. t-SNE mapping of the Ionosphere and Ecoli data sets.

Figure 1 shows the distribution of objects in these two data sets using the t-distributed Stochastic Neighbor Embedding (t-SNE). The Ionosphere data set is composed by two classes. Class 1, represented in red color and grouped in two well defined clusters which are located in the upper and lower section of the space. Class 2, represented in blue color and mainly spread along the mapping space with an identifiable cluster on the right side of the figure. For this case, the CD approach favors the classification of objects of class 2 by assigning more mass to training objects that are located in the central and upper left region, which are clearly *circled by objects of class 1*, thus getting right label assignment even in regions where majority of objects belong to different class. On the other hand, in the Ecoli data set, CD gives more mass to hypothetical noisy objects located away from their normal behavior of its own class (see blue and white objects over the green cluster objects), then negatively affecting the classifier.

5 Conclusions

In this work we introduced the WAF-kNN algorithm, which is a variant of the weighted kNN algorithm but based on the attraction force that exist between two objects. We present two methods of assigning mass to training objects, i.e. *Circled by its own class* (CC) and *Circled by different classes* (CD). For testing purposes 13 known data sets were employed. Comparisons indicate that our proposal obtained better classification results than kNN and is statistically competitive with DWkNN. These results were validated with a non-parametric BSR test.

Acknowledgement. This research was partially supported by CONACYT-Mexico (project FC-2410). The work of Paolo Rosso has been partially funded by the SomEMBED TIN2015-71147-C2-1-P MINECO research project.

References

1. Benavoli, A., Mangili, F., Corani, G., Zaffalon, M., Ruggeri, F.: A Bayesian Wilcoxon signed-rank test based on the Dirichlet process. Proc. 31 Int. Conf. Mach. Learn. **32**, 9 (2014)
2. Bhattacharya, G., Ghosh, K., Chowdhury, A.S.: An affinity-based new local distance function and similarity measure for kNN algorithm. Pattern Recognit. Lett. **33**(3), 356–363 (2012)
3. Carrasco, J., García, S., Del Mar Rueda, M., Herrera, F.: rNPBST: An R Package Covering Non-parametric and Bayesian Statistical Tests. **10334**, 281–292 (2017)
4. Domingos, P.: The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World. Basic Books (2015)
5. Dudani, S.A.: The Distance-Weighted k-Nearest-Neighbor Rule. IEEE Trans. Syst. Man Cybern. **SMC-6**(4), 325–327 (1976)
6. Guru, D.S., Sharath, Y.H., Manjunath, S.: Texture Features and KNN in Classification of Flower Images. Int. J. Comput. Appl. (1), 21–29 (2010)
7. Lam, S.K., Riedl, J.: Shilling recommender systems for fun and profit. Proc. 13th Conf. World Wide Web - WWW '04 p. 393 (2004)
8. López, J., Maldonado, S.: Redefining nearest neighbor classification in high-dimensional settings. Pattern Recognit. Lett. **110**, 36–43 (2018)
9. Mateos-García, D., García-Gutiérrez, J., Riquelme-Santos, J.C.: An evolutionary voting for k-nearest neighbours. Expert Syst. Appl. **43**, 9–14 (2016)
10. Parvinnia, E., Sabeti, M., Zolghadri Jahromi, M., Boostani, R.: Classification of EEG Signals using adaptive weighted distance nearest neighbor algorithm. J. King Saud Univ. - Comput. Inf. Sci. **26**(1), 1–6 (2014)
11. Soucy, P., Mineau, G.: A simple KNN algorithm for text categorization. Proc. 2001 IEEE Int. Conf. Data Min. pp. 647–648 (2001)
12. Tan, S.: Neighbor-weighted K-nearest neighbor for unbalanced text corpus. Expert Syst. Appl. **28**(4), 667–671 (2005)
13. Wilson, D.L.: Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. IEEE Trans. Syst. Man Cybern. **2**(3), 408–421 (1972)
14. Wu, X., Kumar, V., Ross, Q.J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.H., Steinbach, M., H , D.J., Steinberg, D.: Top 10 algorithms in data mining, vol. 14 (2008)
15. Xu, Y., Zhu, Q., Fan, Z., Qiu, M., Chen, Y., Liu, H.: Coarse to fine K nearest neighbor classifier. Pattern Recognit. Lett. **34**(9), 980–986 (2013)
16. Zhang, S., Cheng, D., Deng, Z., Zong, M., Deng, X.: A novel kNN algorithm with data-driven k parameter computation. Pattern Recognit. Lett. **0**, 1–11 (2017)
17. Zhu, Q., Feng, J., Huang, J.: Natural neighbor: A self-adaptive neighborhood method without parameter K. Pattern Recognit. Lett. **80**, 30–36 (2016)