Using Network Analysis to Improve Nearest Neighbor Classification of Non-Network Data

Conference Paper · June 2017

DOI: 10.1007/978-3-319-60438-1_11

4 authors:

Maciej Piernik
Poznan University of Technology
17 PUBLICATIONS **67** CITATIONS
SEE PROFILE

Dariusz Brzezinski
Poznan University of Technology
53 PUBLICATIONS **1,220** CITATIONS
SEE PROFILE

Tadeusz Morzy
Poznan University of Technology
122 PUBLICATIONS **1,145** CITATIONS
SEE PROFILE

Mikolaj Morzy
Poznan University of Technology
92 PUBLICATIONS **665** CITATIONS
SEE PROFILE

Using Network Analysis to Improve Nearest Neighbor Classification of Non-Network Data

Maciej Piernik, Dariusz Brzezinski, Tadeusz Morzy, and Mikolaj Morzy

Institute of Computing Science, Poznan University of Technology
ul. Piotrowo 2, 60–965 Poznan, Poland
maciej.piernik@cs.put.poznan.pl

Abstract. The nearest neighbor classifier is a powerful, straightforward, and very popular approach to solving many classification problems. It also enables users to easily incorporate weights of training instances into its model, allowing users to highlight more promising examples. Instance weighting schemes proposed to date were based either on attribute values or external knowledge. In this paper, we propose a new way of weighting instances based on network analysis and centrality measures. Our method relies on transforming the training dataset into a weighted signed network and evaluating the importance of each node using a selected centrality measure. This information is then transferred back to the training dataset in the form of instance weights, which are later used during nearest neighbor classification. We consider four centrality measures appropriate for our problem and empirically evaluate our proposal on 30 popular, publicly available datasets. The results show that the proposed instance weighting enhances the predictive performance of the nearest neighbor algorithm.

Keywords: classification, instance weighting, nearest neighbors, network analysis, centrality measures

1 Introduction

Instance weighted classification enables data analysts to specify the importance of each training example in order to steer the learning process towards more promising instances. The weights can either be assigned manually by an expert or automatically as a result of some additional process. For these weights to be used by a classifier, it has to possess a certain structure, as not all learning schemes are designed to incorporate weights. One type of learners which has a very straightforward way of using weights is the nearest neighbor classifier [1]. In this method, the class prediction for each new example is based on the classes of the most similar training instances, so instance weights can directly influence the model's predictions.

A domain in which instance weights also play a significant role is network analysis, where the importance of nodes is assessed by centrality measures [2]. As network analysis focuses on relations between nodes rather than their individual

characteristics, centrality measures carry the information about the importance of each node based on its position in the network. Clearly, such information is absent in non-network data where the information about the topology of relationships between nodes is missing.

In this paper, we show how to incorporate network centrality measures into instance-weighted nearest neighbor classification of non-network data. This process consists of three main steps. The first step is the transformation of data into a network using a distance measure and a nearest neighbor approach, so that each node in the network corresponds to one training example and the differences in classes between nodes are expressed as signs of their connections (positive — if from the same class, negative — otherwise). The second step is the assessment of importance of each node in the network. This goal is achieved using one of several centrality measures capable of processing negative ties which we discuss in this paper. The final step is incorporating the centrality values as instance weights into the nearest neighbor classification algorithm. The experiments conducted on 30 datasets show that our weighted nearest neighbor algorithm surpasses the non-weighted version in terms of predictive performance.

In particular, the main contributions of this paper are as follows.

- We propose an automatic instance-weighting algorithm for the nearest neighbor classifier based on network centrality measures.
- We illustrate how to transform a non-network dataset into a network format.
- We discuss several centrality measures and select those which are suitable for instance weighted classification.
- We evaluate our algorithm by performing an experiment involving classification of 30 popular, publicly available datasets.

2 Related Work

The k-nearest neighbors classifier was first proposed by Fix and Hodges in 1951 [1], and has been gaining popularity ever since. In 1976, Dudani [3] proposed a version which weighted the votes of the neighbors according to their distance to the classified example. This method was further refined by Gou et al. [4], where the authors address the problem of sensitivity w.r.t. parameter k. Recently, Samworth [5] proposed an optimal (under certain assumptions) weighting function for nearest neighbor classification of examples with continuous attributes. This approach is based on the ranked similarity of neighbors and is able to automatically select the value of k. These proposals compute the weights based either on attributes or some distance function, but do so dynamically adjusting neighbor weights for each testing example. In contrast, to the best of our knowledge, our proposal is a first attempt to calculate static weights for each training instance based on network analysis.

Network analysis has been a hot research topic for several years now. The area of particular interest to scientists in this field has been the assessment of the importance of nodes in the network. This goal can be achieved using one

of many centrality measures proposed to date [6]. Standard measures, like degree [2], betweenness [2], closeness [2], clustering coefficient [7], or eigenvector [8] centrality, or measures of influence, like Katz [9] or Bonacich [10] centrality, are widely adopted in general network analysis. However, they are designed for unimodal networks (i.e., networks in which all nodes belong to the same category), whereas in this paper we consider multi-modal networks with nodes belonging to multiple categories (where each category corresponds to a single class). Considering the fact that we are expressing the differences in classes between nodes as signs of the weights of their connections, we also have to discard measures designed for networks with exclusively negative ties, like negative degree [11] or h^* centrality [11]. Consequently, we are focusing on measures designed for signed and weighted networks: modified degree, modified clustering coefficient [12], status [13], and PN [11], which will be described in Section 3.

3 Centrality-weighted Nearest Neighbors

Given a dataset of training examples $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_n, y_n)\}$, where $\boldsymbol{x}_i = (x_{i1}, x_{i2}, ..., x_{im})$ is a vector of attribute values and $y_i \in \mathcal{Y}$ is a class label, the task of a classifier is to assign each new unlabeled example $\hat{\boldsymbol{x}}$ to one of the predefined classes $\hat{y} \in \mathcal{Y}$. In case of the nearest neighbor classifier, each new example $\hat{\boldsymbol{x}}$ is classified according to the majority voting of a subset of training examples $\mathcal{N}_{\hat{x}} \subseteq \mathcal{D}$ that are closest to $\hat{\boldsymbol{x}}$, called nearest neighbors of $\hat{\boldsymbol{x}}$:

$$\hat{y} = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} \sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{N}_{\hat{\boldsymbol{x}}}} I(y_i = y), \tag{1}$$

where $I(p)$ equals 1 if predicate p is true and 0, otherwise. The instances can also be weighted, in which case instead of counting examples in each class we simply sum their weights, so the formula from Eq. (1) becomes:

$$\hat{y} = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} \sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{N}_{\hat{\boldsymbol{x}}}} I(y_i = y) w_i, \tag{2}$$

where w_i is the weight of the i-th training example. Notice that if $w_i = 1$ for all training examples $(\boldsymbol{x}_i, y_i) \in \mathcal{D}$, the formulas for weighted and unweighted classifiers are equivalent.

The nearest neighbors of $\hat{\boldsymbol{x}}$ are selected according to some distance function Δ defined on the examples (e.g., euclidean distance). The selection can be carried out in several ways. Common approaches include: choosing k nearest examples (where k is a user-defined value), known as the k-nearest neighbors approach (knn); using only a single closest example (a variant of knn with $k = 1$), known as 1-nearest neighbor (1nn); selecting all examples within a user-defined distance range. In this paper, we will use the most popular knn approach with instance weighting as presented in Eq. (2). We will refer to this algorithm as the weighted k-nearest neighbors classifier (wknn).

Let us now define the distance between two examples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, which we will use to select the nearest neighbors. For this purpose, we will use a measure that is capable of comparing instances on both numerical and categorical attributes [14]. Given that x_{il} is the value of the l-th attribute in example $\boldsymbol{x}_i$, the distance δ between two corresponding attribute values x_{il}, x_{jl} of examples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is defined as follows:

$$\delta(x_{il}, x_{jl}) = \begin{cases} |x_{il} - x_{jl}| & \text{attribute } l \text{ is numerical} \\ I(x_{il} \neq x_{jl}) & \text{attribute } l \text{ is categorical.} \end{cases}$$

Assuming that δ_N gives normalized (rescaled to $\langle 0, 1 \rangle$) values of δ, we define the distance Δ between two examples $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ as an average normalized distance over all attributes:

$$\Delta(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{1}{m} \sum_{l=1}^{m} \delta_N(x_{il}, x_{jl}), \tag{3}$$

where m is the number of attributes.

With all parts of the weighted k-nearest neighbors classifier introduced, let us now describe the proposed process of calculating weights based on centrality measures. The process takes a training dataset $\mathcal{D}$ as input and outputs a vector of weights $\boldsymbol{w}$ where each weight w_i corresponds to one training example $(\boldsymbol{x}_i, y_i) \in \mathcal{D}$. Conceptually, this process consists of three steps: 1) transformation of the input dataset into a network so that each training example is represented by a single node, 2) calculating a selected network centrality measure for each node, 3) assigning the calculated centrality values as instance weights to training examples corresponding to the nodes. This procedure is illustrated in Fig. 1.

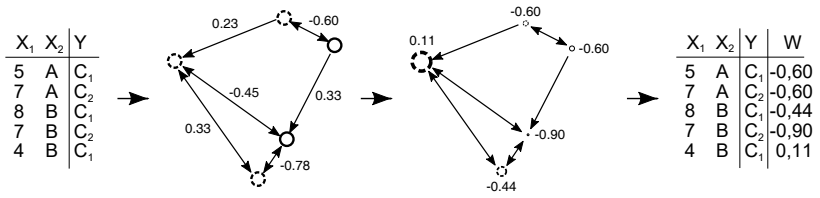


Fig. 1: The process of transforming a dataset into a weighted, signed network (using knn graph with $k_g = 2$), calculating the degree measure, and using the values as weights of instances. Dashed circles represent class C_1, solid circles C_2.

In order to transform the dataset $\mathcal{D}$ into a network, we use an approach inspired by the k-nearest neighbors classifier — the knn graph. First, we calculate the distances between all training examples according to Eq. (3). Next, each training example $(\boldsymbol{x}_i, y_i) \in \mathcal{D}$ becomes a node in the network and is connected with k_g other examples which are closest to it. These connections form directed edges in the network, starting at a given node and ending at its nearest neighbors. As the distances between examples carry potentially valuable information for further processing, we want to include this information in the network in the

form of edge weights. However, because in network analysis edges represent the strength of connections, first, we need to convert the distances into similarities. To achieve this goal, we use a common conversion method [14] and rescale values back to $\langle 0, 1\rangle$, defining the similarity between two examples $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ as:

$$s_N(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{1 - \Delta(\boldsymbol{x}_i, \boldsymbol{x}_j)}{1 + \Delta(\boldsymbol{x}_i, \boldsymbol{x}_j)}.$$

Finally, as each node in the network corresponds to a training example of a certain class, we express this information in the network by adding a sign for each edge. The weight of an edge is positive if the adjacent nodes correspond to training examples from the same class, and negative, otherwise. This gives us the final edge weight between two training examples as:

$$\omega(\boldsymbol{x}_i, \boldsymbol{x}_j) = s_N(\boldsymbol{x}_i, \boldsymbol{x}_j)(2I(y_i = y_j) - 1). \tag{4}$$

After this process is complete, we obtain a network expressed as a graph $G = \langle \mathcal{V}, \mathcal{E}\rangle$ in which $\mathcal{V}$ is a set of nodes, where each node $v_i \in \mathcal{V}$ corresponds to one training example $\boldsymbol{x}_i$, and $\mathcal{E}$ is a set of edges, where each edge $(v_i, v_j, w_{ij}) \in \mathcal{E}$ represents a connection (tie) from node v_i to node v_j with weight w_{ij}. The graph is represented with two matrices: adjacency matrix $\mathbf{A}$, where each element A_{ij} denotes an edge directed from node v_i to v_j, and weight matrix $\mathbf{W}$, where each element W_{ij} denotes the weight of edge A_{ij}. Fig. 1 presents an example of a complete transformation from a training dataset to a network with directed, weighted, and signed ties.

After constructing the network, a selected centrality measure is calculated for every node. Since centrality values will be used as instance weights in wknn classification, the measure has to take into account the signs of the edges. Consequently, one can choose between four different centrality measures: degree, clustering coefficient [12], status [13], and PN [11].

Degree centrality is a classical measure which can be easily adapted to weighted, signed networks. Originally, for a given node v_i, degree centrality d_i is calculated as the number of its connections. In weighted and signed networks it simply becomes a sum of all connection weights of a given node:

$$d_i = \sum_{j=1}^{|\mathcal{V}|} W_{ji}. \tag{5}$$

Because we are interested in the importance of an instance from the perspective of other instances, we only take into account the in-degree of each node.

Clustering coefficient is another classical centrality measure which calculates the level of connectedness of a node's nearest neighborhood. For node $v_i \in \mathcal{V}$, clustering coefficient is the ratio of the number of edges between the nodes which are connected with v_i to the number of all possible edges between them. In unweighted and unsigned networks, clustering coefficient c_i of node v_i is usually calculated according to the Watts and Strogatz definition [7]:

$$c_i = \frac{\sum_{j,k=1}^{|\mathcal{V}|} (A_{ij} A_{ik} A_{jk})}{k_i(k_i - 1)},$$

where k_i is the degree of node v_i. Several versions have been proposed for weighted and signed networks [12], however, given the special meaning of edge signs in our problem, we propose the following definition:

$$c_i = \frac{\sum_{j,k=1}^{|\mathcal{V}|} [(W_{ij} + W_{ik} + W_{jk})(A_{ij}A_{ik}A_{jk})]}{3k_i(k_i - 1)}. \tag{6}$$

The measure produces values between -1 (if all nodes in the neighborhood are fully connected and belong to different classes) and 1 (if all nodes in the neighborhood are fully connected and belong to the same class as node v_i).

Status [13] is a modification of the eignevector centrality, which measures the relative importance of a node w.r.t. the importance of its adjacent nodes using a recursive definition of importance (a node is important if it is adjacent to important nodes). In order to approximate the value of the status measure we rely on the power iteration method defined as follows:

$$\boldsymbol{s}^l = \frac{\mathbf{W}\boldsymbol{s}^{l-1}}{max(\mathbf{W}\boldsymbol{s}^{l-1})}, \tag{7}$$

where $\boldsymbol{s}^l$ is a vector of statuses for all nodes in iteration $l > 0$ and $\boldsymbol{s}^0 = \mathbf{1}$.

Finally, PN [11] is a measure designed specifically for networks with signed ties and is defined as follows:

$$\boldsymbol{pn} = \left(\mathbf{I} - \frac{1}{2|\mathcal{V}| - 1}\mathbf{M}\right)^{-1}\mathbf{1}. \tag{8}$$

In the equation, $\boldsymbol{pn}$ is a vector of PN values for all nodes and $\mathbf{M} = \mathbf{P} - 2\mathbf{N}$, where $\mathbf{P}$ and $\mathbf{N}$ denote two matrices containing positive and negative ties, respectively.

In the following section, we experimentally verify the predictive performance of wknn with instances weighted according to the described centrality measures.

4 Experiments

4.1 Experimental Setup

The goal of this paper is to assess the effect of centrality instance weighting on the predictive performance of the nearest neighbor classifier. For this purpose, we compare the knn classifier with instances weighted:

- identically (equivalent of unweighted knn; uniform),
- randomly according to a uniform distribution between 0 and 1 (random),
- using clustering coefficient (cluster),
- based on the in-degree (degree),
- based on PN (PN),
- using status (status),
- using the centrality measure with the highest validation score (bestCV).

The first two approaches serve as baselines of traditional (unweighted) knn and random instance weighting; the remaining approaches test the usefulness of the proposed data transformation and centrality measures.

To test the proposed solution we first divided each dataset into a training set and a holdout test set consisting of 50% of the original data. Next, we performed 5×2-fold cross-validation [15] on the training set to select the best k for knn. This way, we tune knn on each dataset to have the best possible performance without instance weighting and make the comparison more challenging. After setting k for each dataset, we performed 5×2-fold cross-validation on the training set once again, but this time to select the k_g parameter used to create the knn graph for each centrality measure. During parameter tuning, we additionally highlight the centrality measure that achieved the best mean cross-validation score as bestCV. Finally, each model was evaluated on a holdout test set.[1] Fig. 2 depicts the experimental procedure.

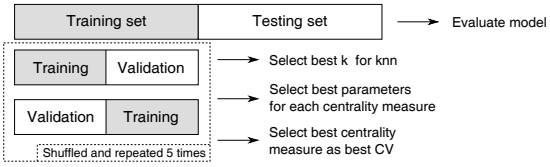


Fig. 2: Experimental procedure.

Model selection as well as evaluation on the test set were performed using the κ statistic [15]. In the context of classification, the κ statistic (or Cohen's κ) compares the accuracy of the tested classifier with that of a chance classifier: $\kappa = \frac{p_0-p_c}{1-p_c}$, where p_0 is the accuracy of the tested classifier and p_c is the accuracy of a chance classifier [15]. The κ statistic can achieve values from $(-\infty; 1]$, where zero means that the tested model is no better than a chance classifier and values above/below zero indicate how much better/worse the model performs compared to chance. We selected κ as the evaluation measure in this study, because it is suitable for datasets that suffer from class-imbalance and can be applied to binary as well as multi-class problems.

4.2 Datasets

In our experiments, we used 30 datasets with various numbers of classes, imbalance ratios, and containing nominal as well as numeric attributes. All of the used datasets are publicly available through the UCI machine learning repository [16]. Table 1 presents the main characteristics of each dataset.

[1] Source code in R and reproducible test scripts available at: http://www.cs.put.poznan.pl/dbrzezinski/software.php

Table 1: Characteristic of datasets

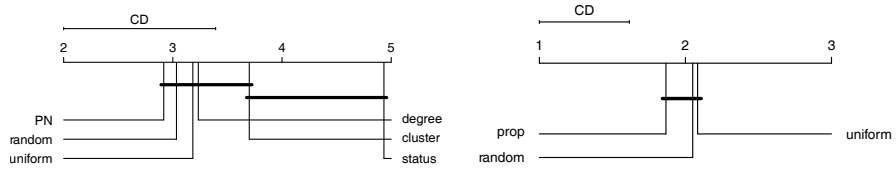
Dataset	Inst.	Attr.	Classes	Dataset	Inst.	Attr.	Classes
balance-scale	625	4	3	monks-1	556	6	2
breast-cancer	699	9	2	monks-2	601	6	2
car-evaluation	1,728	6	4	monks-3	554	6	2
cmc	1,473	9	3	pima	768	8	2
credit-screening	690	15	2	promoters	106	57	2
dermatology	366	34	6	sonar	208	60	2
ecoli	336	7	8	spect	267	22	2
glass	214	9	6	statlog-australian	690	14	2
haberman	306	3	2	statlog-heart	270	13	2
heart-disease	303	13	5	tae	151	5	3
hepatitis	155	19	2	tic-tac-toe	958	9	2
house-votes-84	435	16	2	vowel-context	990	10	11
image-segmentation	2,310	19	7	wine	178	13	3
ionosphere	351	34	2	yeast	1,484	8	10
iris	150	4	3	zoo	101	16	7

The datasets were selected based on their availability and popularity among other studies. As can be seen in Table 1, the testbed consists of binary as well as multiclass problems with a wide range of instance and attribute counts.

4.3 Results

Table 2 presents evaluation results for the analyzed instance weighting schemes. Values marked in bold represent the best result on a given dataset.

We can notice that there is no clear winner in this comparison. Additionally, it is worth highlighting that whenever knn without instance weighting (uniform) achieves best results, usually one of the centrality measures is equally accurate.



(a) Centrality measures separately (b) Measure chosen during validation

Fig. 3: Performance ranking of instance weighting schemes. Measures that are not significantly different according to the Nemenyi test (at $\alpha = 0.05$) are connected.

To verify the significance of the observed differences, we performed the non-parametric Friedman test [15]. The null-hypothesis of the Friedman test (that there is no difference between the performance of all the tested instance weighting schemes) was rejected. To verify which instance weighting performs better than the other, we computed the critical difference (CD) chosen by the Nemenyi post-hoc test [15] at $\alpha = 0.05$. Fig. 3 depicts the results of the test by connecting the groups of measures that are not significantly different.

Table 2: Kappa statistic for knn with different instance weighting schemes.

Dataset	uniform	random	degree	cluster	status	PN	bestCV
balance-scale	0.83	0.80	0.79	0.83	0.47	**0.83**	**0.83**
breast-cancer	0.92	**0.93**	0.92	0.92	0.92	0.92	0.92
car-evaluation	**0.80**	0.74	0.45	**0.80**	0.34	0.76	**0.80**
cmc	0.14	0.11	0.15	0.14	0.12	**0.16**	**0.16**
credit-screening	0.69	0.67	**0.72**	0.69	0.63	0.69	0.69
dermatology	**0.95**	**0.95**	0.94	0.88	0.86	0.94	0.88
ecoli	**0.76**	0.74	0.73	0.73	0.70	**0.76**	**0.76**
glass	0.53	0.52	**0.60**	0.52	0.54	0.55	0.55
haberman	0.03	0.04	**0.16**	0.11	0.00	0.09	0.09
heart-disease	0.22	**0.26**	0.23	0.19	0.21	0.22	0.21
hepatitis	0.37	0.39	0.30	0.37	0.00	**0.48**	0.37
house-votes-84	0.81	**0.83**	**0.83**	0.81	0.80	0.81	0.81
image-segmentation	**0.96**	**0.96**	**0.96**	**0.96**	**0.96**	**0.96**	**0.96**
ionosphere	**0.81**	0.78	0.77	0.70	0.77	0.70	0.70
iris	0.92	0.92	**0.94**	0.92	0.90	0.92	**0.94**
monks-1	0.82	0.81	0.66	0.82	0.40	**0.86**	0.82
monks-2	-0.02	**0.06**	0.04	0.04	0.00	-0.01	0.04
monks-3	**0.96**	0.93	0.85	**0.96**	0.46	0.95	**0.96**
pima	0.34	0.35	0.34	0.34	0.24	**0.37**	0.34
promoters	0.46	**0.50**	0.42	0.46	0.42	0.42	0.42
sonar.all	**0.71**	**0.71**	**0.71**	**0.71**	**0.71**	**0.71**	**0.71**
spect	**0.44**	0.39	**0.44**	0.36	0.43	0.40	**0.44**
statlog-australian	**0.68**	0.66	0.67	**0.68**	0.59	**0.68**	**0.68**
statlog-heart	0.56	0.58	**0.59**	0.35	0.52	0.56	0.56
tae	0.28	**0.30**	0.28	0.28	**0.30**	0.28	**0.30**
tic-tac-toe	**0.97**	0.92	0.61	**0.97**	0.29	0.92	**0.97**
vowel-context	**0.91**	**0.91**	**0.91**	**0.91**	**0.91**	**0.91**	**0.91**
wine	**0.93**	0.91	**0.93**	**0.93**	0.83	**0.93**	**0.93**
yeast	0.46	**0.46**	0.42	0.46	0.35	0.46	0.46
zoo	0.95	**0.97**	**0.97**	0.95	**0.97**	**0.97**	0.95

We can notice that the status and cluster measures perform poorly compared to the remaining weighting schemes. PN and degree, on the other hand, present promising results, with PN achieving the best average rank in the Friedman test. Interestingly, if we select the best centrality measure during model selection, the result is not better than when using PN alone. This may suggest that some of the analyzed measures may be prone to overfitting or generalize poorly with more data. This in turn may be an interesting guideline when attempting to propose new network-based measures for instance weighting.

5 Conclusions

In this paper, we presented a new method of weighting instances in the nearest neighbor classifier based on network analysis. The approach relies on transforming the dataset into a weighted and signed network, calculating the centrality of each node, and later using this information during classification. We discussed the transformation process as well as several centrality measures capable of dealing with weighted and signed connections. The experiments performed on 30 popular datasets show that our weighted approach can perform favorably to the unweighted version.

As future research, we plan on experimenting with alternative ways of constructing networks from non-network data, as well as proposing new centrality measures dedicated for the stated problem. Regarding centrality measures, an extensive discussion of their properties w.r.t. classification would also constitute a very interesting line of future research. Furthermore, as evidenced by our experiments, the performance of measures varies with different datasets, therefore, it would be interesting to verify which measures suit which data characteristics better. Finally, generalizing our solution to other classification algorithms would be of high interest as it would make our proposal a generic approach to enhancing classifier performance.

Acknowledgments. This research is partly funded by the Polish National Science Center under Grant No. 2015/19/B/ST6/02637. D. Brzezinski acknowledges the support of an FNP START scholarship.

References

1. Fix, E., Hodges, J.L.: Discriminatory analysis, nonparametric discrimination: Consistency properties. US Air Force School of Aviation Medicine **Technical Report 4**(3) (1951) 477+
2. Freeman, L.C.: Centrality in social networks conceptual clarification. Social Networks **1**(3) (1978) 215–239
3. Dudani, S.A.: The Distance-Weighted k-Nearest-Neighbor Rule. IEEE Transactions on Systems, Man, and Cybernetics **SMC-6**(3) (1976) 325–327
4. Gou, J., Du, L., Zhang, Y., Xiong, T.: A New Distance-weighted k-nearest Neighbor Classifier. J. of Information and Computational Science **9**(6) (2012) 1429–1436
5. Samworth, R.J.: Optimal weighted nearest neighbour classifiers. The Annals of Statistics **40**(5) (2012) 2733–2763
6. Kaur, M., Singh, S.: Analyzing negative ties in social networks: A survey. Egyptian Informatics Journal **17**(1) (2016) 21–43
7. Watts, D.J., Strogatz, S.H.: Collective dynamics of'small-world'networks. Nature **393**(6684) (1998) 409–10
8. Bonacich, P.: Factoring and weighting approaches to status scores and clique identification. Journal of Mathematical Sociology **2**(1) (1972) 113–120
9. Katz, L.: A new status index derived from sociometric analysis. Psychometrika **18**(1) (1953) 39–43
10. Bonacich, P.: Power and Centrality: A Family of Measures. American Journal of Sociology **92**(5) (1987) 1170–1182
11. Everett, M., Borgatti, S.: Networks containing negative ties. Social Networks **38** (7 2014) 111–120
12. Costantini, G., Perugini, M.: Generalization of clustering coefficients to signed correlation networks. PLOS ONE **9**(2) (02 2014) 1–10
13. Bonacich, P., Lloyd, P.: Calculating status with negative relations. Social Networks **26**(4) (2004) 331–338
14. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison Wesley (2005)
15. Japkowicz, N., Shah, M.: Evaluating Learning Algorithms: A Classification Perspective. Cambridge University Press (2011)
16. Lichman, M.: UCI machine learning repository (2013)