Nearest Centroid Classifier Based on Information Value and Homogeneity

Mehmet Hamdi Özçelik[1,2(✉)] and Serol Bulkan[2]

[1] Applied Analytics AA, İstanbul, Türkiye
hamdi.ozcelik@appliedanalytics.com.tr, hamdiozcelik@marun.edu.tr
[2] Department of Industrial Engineering, Marmara University, İstanbul, Türkiye
sbulkan@marmara.edu.tr

Abstract. The aim of this paper is to introduce a novel classification algorithm based on distance to class centroids with weighted Euclidean distance metric. Features are weighted by their predictive powers and in-class homogeneities. For predictive power, information value metric is used. For in-class homogeneity different measures are used. The algorithm is memory based but only the centroid information needs to be stored. The experimentations are carried at 45 benchmark datasets and 5 randomly generated datasets. The results are compared against Nearest Centroid, Logistic Regression, K-Nearest Neighbors and Decision Tree algorithms. The parameters of the new algorithm and of these traditional classification algorithms are tuned before comparison. The results are promising and has potential to trigger further research.

Keywords: Machine Learning · Classification · Similarity Classifier · Nearest Centroid · Information Value

1 Introduction

As one of the most important theorems in statistical learning, the no free lunch theorem [1] states that the performance of an algorithm could be better than others at some problems and worse at some others, which leads to some type of equivalence among algorithms. Due to the natural differences among classification problems many different classifiers are designed so far. In this paper we introduce a novel variant of Nearest Centroid (NC) classifier [2].

In our new algorithm, the distance measure is a weighted Euclidean metric where the predictive power of each feature and homogeneity of each feature at each class are used to determine weights. Information Value (IV) metric is selected as the metric showing the predictive power of features. For measuring homogeneity, mean absolute deviation, standard deviation, variance and coefficient of variance metrics are used. The choice of these metrics became a parameter of our algorithm and is used at the tuning phase.

A binary classification model provides predicted binary classes, predicted probabilities of each class or rank information of these probabilities and different performance measures are used for evaluation [3]. We have used accuracy measure to compare the

Z. Şen et al. (Eds.): IMSS 2023, LNME, pp. 36–45, 2024.
https://doi.org/10.1007/978-981-99-6062-0_5

performance of the algorithm. For benchmarking, we used 50 different datasets and 4 different algorithms, namely Nearest Centroid (NC), Logistic Regression (LR), K-Nearest Neighbours (KNN) and Decision Tree (DT). The new algorithm outperformed NC at 43 datasets, LR at 8 datasets, KNN at 17 datasets and DT at 15 datasets. It was the best classifier at 5 datasets with respect to accuracy measure.

2 Related Research

Nearest Centroid Classifier [2] is a memory-based classifier which is simple and fast. It stores the centroid information of each class and then makes distance calculations for each new instance. The nearest centroids' class is assigned as the predicted class of that instance; therefore, the algorithm is also called as Minimum Distance Classifier (MDC).

Instead of class centroids, some instances at the training dataset could be chosen to represent that class [4]. These instances are called as prototypes.

Nearest shrunken centroid classifier [5] is another variant of Nearest Centroid algorithm. It shrinks each of the class centroids toward the overall centroid for all classes by an amount called threshold. After the centroids are shrunken, a new instance is classified by nearest centroid rule, but using shrunken class centroids.

Nearest Centroid Classifier, K-Means Clustering [6] and K-Nearest Neighbours (KNN) [7] algorithms are closely related with each other since they consider centroid information. KNN and its variants [8] are used for both classification and regression problems. Using weights for centroid based distances are common and so far, different measures are used. For example, Gou et.al. [9] proposed an algorithm which captures both the proximity and the geometry of k-nearest neighbours. At another recent research, Elen et.al. [10] proposes a new classifier by addressing the noise issue at classification by introducing standardized variable distances.

Baesens et.al. [11] compared the performance of different algorithms over some datasets and two of them are in our list, namely "Australian" and "credit-g (German Credit)". For the Australian dataset, the accuracy of our new algorithm is the third best one, after the algorithms "C4.5rulcs dis" and "C4.5dis". For the German Credit dataset, the accuracy of our new algorithm (76.40) is above the best algorithm tested (LDA with a value of 74.6). Lessmann et.al. [12] updated this research by testing various classification algorithms over 8 datasets.

The novelty of our algorithm is at the usage of both information value and homogeneity metrics for weighting the distance calculation.

3 Methods

3.1 Information Value

Information Value (IV) metric is used to determine the predictive power of each feature. Its calculation is based on "Weight of Evidence" values [13]. The Weight of Evidence (WOE) of a feature value is defined as the natural logarithm of the ratio of the share of

one class in a given value over the share of other class as shown in Eq. 1.

$$WoE(X_i = X_{ij}) = ln\left(\frac{share\ of\ responses\ of\ that\ category\ at\ all\ responses}{share\ of\ nonresponses\ of\ that\ category\ at\ all\ nonresponses}\right) \quad (1)$$

Equation 2 gives the computation of Information Value over the Weight of Evidence values. The differences of percentage shares of classes are used as weights at the summation.

$$IV(X) = \sum_{j=1}^{V} (Distribution\ of\ class\ 1 - Distribution\ of\ class\ 0) \cdot WoE_j \quad (2)$$

Information Value could be calculated only for categoric variables. For that reason, we split continuous variables into 10 equal sized bins and then made the calculation.

Both Information Value and Weight of Evidence metrics are widely used at data analysis for credit scoring at the banking sector.

3.2 Homogeneity Metrics

For each feature, the following sparsity metrics are computed to represent the inverse of homogeneity within each class:

1. Mean absolute deviation
2. Standard deviation
3. Variance
4. Coefficient of variation

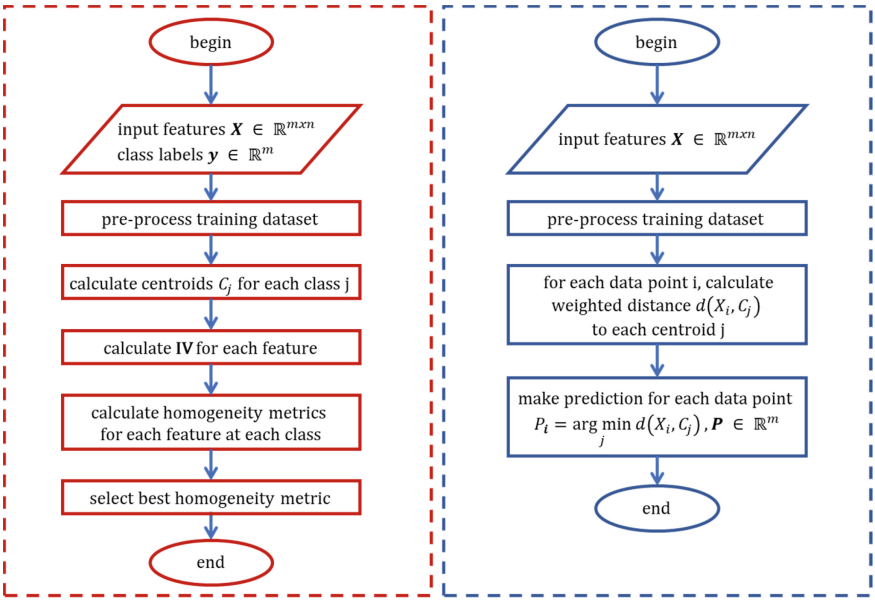
All of them are used at the training phase to fit classifier model into dataset. As the last step the one with the highest accuracy is marked as the selected homogeneity metric. At the tuning phase of the algorithm, in addition to these 4 choices, using no metric for homogeneity is also checked and it is selected when its accuracy is highest. Therefore, there were 5 alternatives for homogeneity.

3.3 The Algorithm

At the training phase, first, the centroids of each class, information values of each feature and homogeneity values of each feature at each class are computed. Then, for the given dataset, the best homogeneity metric is determined by running the scoring at the training dataset using the accuracy metric. Figure 1 (a) depicts the flow of training phase.

At the scoring phase, for each new instance, the Euclidean distance to each class centroid is calculated where the weights are determined as the division of the feature's Information Value to the selected sparsity metric. The class with the minimum weighted distance becomes the predicted class for that instance. Figure 1 (b) depicts the flow of scoring phase.

The algorithm is a memory-based algorithm but it does not require storing the instances at the memory, instead only class summaries (i.e., centroid vectors) need to be stored. We used only datasets of binary classification problems but the algorithm could also be used for multiclass classification problems.



(a) Training phase (b) Scoring phase

Fig. 1. Algorithm flowcharts

4 Experimentation and Results

4.1 Setup

All experiments are performed on a PC equipped with 4 Core Intel i7 11th Gen CPU at 2.80 GHz and 32 GB RAM running Microsoft Windows 11 Pro, Anaconda 3, Conda 22.9.0, Jupyter-notebook 6.4.12 and Python 3.9.13.

4.2 Datasets

We used OpenML [14] machine learning repository which is a public online platform built for scientific collaboration. We also used its Python API [15] to access datasets at the repository. We downloaded 45 binary classification problem datasets using this API. We also generated 5 synthetic datasets for classification via "make_classification" function of Scikit-Learn library [16] and named with a prefix "random". Table 1 shows the datasets used at the experiments.

The second column ("data_id") refers to the unique identifier at OpenML repository. Table 1 also lists the number of instances and features at each dataset. Since we pre-processed data, the numbers of features are changed and the final number of features are given at the last column of the table.

Table 1. List of datasets

Dataset	data_id	İnstances	features	final features
Adult	179	48842	13	109
adult-census	1119	32561	13	97
Australian	40981	690	13	34
bank-marketing	1461	45211	15	42
banknote-authentication	1462	1372	3	4
blood-transfusion-service-center	1464	748	3	4
breast-cancer	13	286	8	42
breast-w	15	699	8	9
Churn	40701	5000	19	29
Click_prediction_small	1220	39948	8	9
climate-model-simulation-crashes	1467	540	19	20
credit-approval	29	690	14	38
credit-g	31	1000	19	50
Diabetes	37	768	7	8
eeg-eye-state	1471	14980	13	14
Electricity	151	45312	7	13
Elevators	846	16599	17	18
heart-c	49	303	12	18
heart-statlog	53	270	12	13
hill-valley	1479	1212	99	100
İlpd	1480	583	9	10
İonosphere	59	351	33	34
jm1	1053	10885	20	21
kc1	1067	2109	20	21
kc2	1063	522	20	21
kc3	1065	458	38	39
kr-vs-kp	3	3196	35	38
MagicTelescope	1120	19020	9	10
mozilla4	1046	15545	4	5
Musk	1116	6598	166	267
ozone-level-8h	1487	2534	71	72
pc1	1068	1109	20	21

(*continued*)

Table 1. (*continued*)

Dataset	data_id	İnstances	features	final features
pc2	1069	5589	35	36
pc3	1050	1563	36	37
pc4	1049	1458	36	37
PhishingWebsites	4534	11055	29	38
Phoneme	1489	5404	4	5
qsar-biodeg	1494	1055	40	41
random1	−1	5000	19	20
random2	−2	5000	19	20
random3	−3	5000	19	20
random4	−4	5000	19	20
random5	−5	5000	19	20
Scene	312	2407	298	299
Sick	38	3772	28	31
Sonar	40	208	59	60
Spambase	44	4601	56	57
steel-plates-fault	1504	1941	32	33
tic-tac-toe	50	958	8	18
Wdbc	1510	569	29	30

4.3 Pre-Processing

The following pre-processing steps are applied to all datasets:

- Splitting: Each dataset is randomly divided into training and test datasets where the share of test dataset set to 25%.
- Null Value Imputation: Null values were replaced by zero values.
- Winsorization: For numeric features of the training dataset, 5% cut-off values are calculated from each end. The values beyond these limits were replaced by the cut-off values. Winsorization is applied only when the number of unique values of the feature is greater than 60.
- Min-Max Scaling: All numeric feature values are proportionally scaled into [0,1] range.
- One-hot encoding: For each distinct value of a categoric feature, a new flag variable is created.

4.4 Tuning

For a healthy comparison, the hyperparameters of Logistic Regression, KNN and Decision Tree algorithms are tuned with the options shown in Table 2. For each dataset, the tuning is made with an exhaustive search over these parameters via the "GridSearchCV" function of Scikit-learn library [16].

Table 2. Parameters used at the tuning

Algorithm	Parameter	Values
Logistic Regression	Penalty	l1','l2','elasticnet','none'
	C	0.01, 0.1, 1.0, 10, 100
	Solver	newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'
KNN	n_neighbors	5,10,30
	Weights	uniform','distance'
Decision Tree	min_samples_leaf	30, 50, 100
	Criterion	gini', 'entropy', 'log_loss'

4.5 Results

Table 3 shows the accuracy values of 5 algorithms over the test datasets. The proposed algorithm **outperforms all other** four algorithms at 5 datasets and **shares the first place** at another 3 datasets. The algorithm has a better accuracy score than nearest centroid at 43 datasets. It was better than Logistic Regression, KNN and Decision Tree algorithms 8, 17 and 15 datasets respectively.

We calculated the correlation coefficients among the accuracy values of our algorithm with the accuracy values of others. Our algorithms accuracy values over test datasets are correlated with the ones of Nearest Centroid, Logistic Regression, KNN and Decision Trees by 0.28, 0.25, 0.30 and 0.35 respectively.

Table 3. Accuracy of algorithms at test datasets

Dataset	NC	LR	KNN	DT	New Algorithm
Adult	0.7236	**0.8507**	0.8287	0.8476	0.8394
adult-census	0.7255	**0.8435**	0.8322	0.8430	0.8348
Australian	0.8786	0.8624	0.8590	0.8416	**0.8902**
bank-marketing	0.7282	**0.9017**	0.8913	0.8991	0.8922

(*continued*)

Table 3. (*continued*)

Dataset	NC	LR	KNN	DT	New Algorithm
banknote-authentication	0.8367	0.9918	**0.9988**	0.9493	0.8542
blood-transfusion-service-center	0.7380	0.7561	0.7626	**0.7733**	0.7594
breast-cancer	0.6806	0.7028	0.6667	0.7056	**0.7222**
breast-w	0.9657	0.9646	0.9749	0.9280	**0.9771**
Churn	0.6776	0.8648	0.8827	**0.9400**	0.8408
Click_prediction_small	0.5586	**0.8320**	0.8014	0.8239	0.7548
climate-model-simulation-crashes	0.7407	0.8963	0.9126	**0.9185**	0.8889
credit-approval	0.8555	0.8509	0.8335	0.8474	**0.8728**
credit-g	0.7360	0.7304	0.7304	0.7096	**0.7640**
Diabetes	0.7708	**0.7740**	0.7563	0.7563	0.7448
eeg-eye-state	0.5848	0.6529	**0.9503**	0.7930	0.6326
Electricity	0.7039	0.7579	0.8496	**0.8531**	0.7383
Elevators	0.7499	**0.8760**	0.8097	0.8343	0.7629
heart-c	0.8026	**0.8474**	0.8132	0.7237	0.8026
heart-statlog	0.7794	**0.8088**	0.7853	0.6735	**0.8088**
hill-valley	0.4455	**0.9208**	0.5116	0.5201	0.4785
Ilpd	0.6233	**0.7288**	0.6890	0.6836	0.6918
Ionosphere	0.7159	**0.9000**	0.8159	0.8932	0.8409
jm1	0.7241	**0.8168**	0.8018	0.8048	0.7535
kc1	0.7727	0.8477	**0.8481**	0.8356	0.7973
kc2	0.8321	**0.8519**	0.8366	**0.8519**	0.8321
kc3	0.8000	0.8852	0.8957	**0.9026**	0.8348
kr-vs-kp	0.8511	**0.9730**	0.9602	0.9692	0.8836
MagicTelescope	0.7586	0.7898	0.8420	**0.8451**	0.7819
mozilla4	0.7335	0.8502	0.8920	**0.9420**	0.8004
Musk	0.7291	**1.0000**	0.9842	0.9565	0.9988
ozone-level-8h	0.6877	0.9391	**0.9423**	0.9252	0.9117
pc1	0.7950	0.9266	**0.9403**	0.9317	0.8921
pc2	0.8777	0.9957	**0.9963**	0.9963	0.0043

(*continued*)

Table 3. (*continued*)

Dataset	NC	LR	KNN	DT	New Algorithm
pc3	0.7647	0.8859	**0.8885**	0.8788	0.8721
pc4	0.7699	**0.9134**	0.8740	0.8827	0.8849
PhishingWebsites	0.9045	0.9263	**0.9581**	0.9336	0.9157
Phoneme	0.7365	0.7504	**0.8934**	0.8278	0.7757
qsar-biodeg	0.7803	**0.8598**	0.8311	0.7955	**0.8598**
random1	0.8272	0.8453	**0.9261**	0.8587	0.8208
random2	0.7568	0.7806	**0.9026**	0.8189	0.8312
random3	0.9000	0.9437	**0.9550**	0.9064	0.9216
random4	0.7496	0.7770	**0.8978**	0.8200	0.7736
random5	0.7592	0.8010	**0.8885**	0.8198	0.7640
Scene	0.7292	**0.9864**	0.9140	0.9535	0.9153
Sick	0.7614	**0.9578**	0.9565	0.9659	0.9502
Sonar	0.6923	0.7808	**0.8346**	0.7231	0.6346
Spambase	0.8888	**0.9361**	0.9333	0.9003	0.9253
steel-plates-fault	0.6461	**1.0000**	0.9835	**1.0000**	**1.0000**
tic-tac-toe	0.7000	0.9792	**0.9992**	0.7817	0.7292
Wdbc	**0.9790**	0.9678	0.9566	0.8825	0.9650

5 Discussion and Conclusion

Weight of Evidence and Information Value metrics are commonly used at banking to predict credit defaults and in our experiments, there were three datasets that are in credit risk domain, namely “Australian”, “credit-approval” and “credit-g (German Credit)”. In all of these three datasets, the new algorithm was better than all others. This could be further investigated using additional datasets from credit risk domain.

The new algorithm has clear superiority over Nearest Centroid algorithm and comparable performance to other classic algorithms. It could be improved by considering other characteristics such as the size of the classes.

To improve the performance of the algorithm various alternatives may be considered. For example, instead of Information Value, another measure of predictive performance such as variable importance could be used. Similarly, at the distance calculation, Euclidean formula could be replaced by other distance metrics such as Manhattan.

Over 50 datasets, the new algorithm outperforms the benchmark algorithms at 5 datasets and shares the first place at another 3 datasets. Therefore, the new algorithm is comparable to well-known algorithms and it should be enhanced further.

6 Declaration of Competing Interest

The authors declare that there is no conflict of interest identified in this study.

Acknowledgements. The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

References

1. Wolpert, D.H.: The supervised learning no-free-lunch theorems. Soft Comput. Ind. 25–42 (2002)
2. Hastie, T., Tibshirani, R., Friedman, J.H.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, vol. 2, p. 670. Springer, New York (2009)
3. Shmueli, G.: To explain or to predict? Stat. Sci. **25**(3), 289–310 (2010)
4. Kuncheva, L.I.: Prototype classifiers and the big fish: the case of prototype (instance) selection. IEEE Syst. Man Cybern. Mag. **6**(2), 49–56 (2020)
5. Tibshirani, R., Hastie, T., Narasimhan, B., Chu, G.: Diagnosis of multiple cancer types by shrunken centroids of gene expression. Proc. Natl. Acad. Sci. **99**(10), 6567–6572 (2002)
6. Hartigan, J.A., Wong, M.A.: Algorithm AS 136: a k-means clustering algorithm. J. Roy. Stat. Soc. Ser. C (Appl. Stat.) **28**(1), 100–108 (1979)
7. Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theory **13**(1), 21–27 (1967)
8. Alpaydin, E.: Voting over multiple condensed nearest neighbors. In: Aha, D.W. (eds.) Lazy Learning, pp. 115–132. Springer, Dordrecht (1997). https://doi.org/10.1007/978-94-017-2053-3_4
9. Gou, J., et al.: A representation coefficient-based k-nearest centroid neighbor classifier. Expert Syst. Appl. **194**, 116529 (2022)
10. Elen, A., Avuçlu, E.: Standardized Variable Distances: a distance-based machine learning method. Appl. Soft Comput. **98**, 106855 (2021)
11. Baesens, B., Van Gestel, T., Viaene, S., Stepanova, M., Suykens, J., Vanthienen, J.: Benchmarking state-of-the-art classification algorithms for credit scoring. J. Oper. Res. Soc. **54**, 627–635 (2003)
12. Lessmann, S., Baesens, B., Seow, H.V., Thomas, L.C.: Benchmarking state-of-the-art classification algorithms for credit scoring: an update of research. Eur. J. Oper. Res. **247**(1), 124–136 (2015)
13. Siddiqi, N.: Intelligent Credit Scoring: Building and Implementing Better Credit Risk Scorecards, pp.186–197. Wiley (2017)
14. Vanschoren, J., Van Rijn, J.N., Bischl, B., Torgo, L.: OpenML: networked science in machine learning. ACM SIGKDD Explor. Newslett. **15**(2), 49–60 (2014)
15. Feurer, M., et al.: OpenML-Python: an extensible Python API for OpenML. J. Mach. Learn. Res. **22**(1), 4573–4577 (2021)
16. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)