Weighted Hamming Metric and KNN Classification of Nominal-Continuous Data

Aleksander Denisiuk[0000−0002−7501−7048]

University of Warmia and Mazury in Olsztyn
ul. Słoneczna 54, 10-710 Olsztyn, Poland
denisiuk@matman.uwm.edu.pl

Abstract. The purpose of the article is to develop a new metric learning algorithm for combination of continuous and nominal data. We start with Euclidean metric for continuous and Hamming metric for nominal part of data. The impact of specific feature is modeled with corresponding weight in the metric definition. A new algorithm for automatic weights detection is proposed. The weighted metric is then used in the standard knn classification algorithm. Series of numerical experiments show that the algorithm can successfully classify raw, non-normalized data.

Keywords: KNN classification · weighted Hamming metric · nominal-continuous data · metric learning.

1 Introduction

Similarity is important concept of data science. Most algorithms operate on groups of similar data, and their results strongly depend on how we define similarity. Mathematical concept that can be used to model similarity is metric. Assuming that the similarity (metric) is hidden in the data, we get the problem of determining the metric, i.e. *metric learning*.

The purpose of this article is further development of the metric learning technics suggested in [4] for combinations of nominal-continuous data. Namely, instead of unsupervised learning and clustering investigated in [4], here we consider supervised learning and the classification problem.

We do not assume that any additional structure on a dataset is a-priory known, so we use the Euclidean metric on continuous and the Hamming distance on nominal part of data. These metrices provide the most common and straightforward way for measuring distances [8].

The main assumption is that each feature has different impact to the structure of classes. We model it with appropriate multipliers in metric (1). So, the problem is to define these unknown multipliers. We do it by minimizing the total intra-class squared distance.

As a possible application of our technics we show that the standard knn-classification algorithm can be improved by using the weighted metric. Let us, however, note that the proposed approach can be used in any algorithm for analysis of nominal-continuous data that is based on metric.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_31

The paper is organized as follows. In the section 2 we give a brief survey of related works. The algorithm of weights detection is the content of the section 3. The results of numerical experiments are presented in the section 4. Some final conclusions and possible directions for the future work are given in the section 5.

2 Related Works

The metric learning is actively studied in recent years. However, most papers consider either continuous or nominal data. Sometimes nominal data are embedded into continuous space, but this embedding as a rule is arbitrary. In this article we consider nominal and continuous data together in a natural way: the Euclidean metric on continuous part and the Hamming metric on nominal one. Data from each part affects the weights on the other part as well.

One of the first papers in metric learning of continuous data is [17], where a projected gradient descent algorithm for Mahalanobis distance learning is suggested. Two stochastic methods, the neighborhood component analysis and the large margin nearest neighbors were introduced in [6] and [16] respectively. The above methods got further development in later years, see for instance [15].

The authors of [14] used the support vector machine approach to develop an algorithm for the Mahalanobis distance learning. Other approach, the information-theoretic metric learning was introduced in [3].

Let us also mention [7], where the weighted Euclidean distance was considered. In our paper we extend this model to combinations of nominal and continuous data. We refer to works [9] and [2] for more details and references.

The bibliography of metric learning for nominal data is not so extensive. Most of papers assume some additional structure given on nominal data and develop an algorithm for learning of appropriate metric, e.g. tree-editing metric or string-editing metric, see the survey [2].

The Hamming distance itself was considered in [12, 18]. The authors defined and optimized projections from continuous features into product of binary sets with the standard Hamming distance. This differs from our context: we assume that the data already have nominal (not only binary) features.

The general space of nominal-continuous data was considered in [4] in context of non-supervised learning. This article is the direct predecessor of our work.

3 Determining the Weights

We consider the data $\mathbb{X} = \{ \mathbf{X}_1, \dots, \mathbf{X}_M \}$ of M records. Each record consists of two parts $\mathbf{X}_i = (X_i, Y_i)$, where $X_i = (x_i^1, \dots, x_i^n) \in \mathbb{R}^n$ are the continuous data, and $Y_i = (y_i^1, \dots, y_i^m)$ are the nominal data, $i = 1, \dots, M$. We assume that $\mathbb{X}$ is divided into c classes, $\mathbb{X} = C_1 \cup \dots \cup C_c$, where $c < M$. These classes will be used for learning. The determined weights are used for classification of new records.

The Hamming metric on the set of nominal data is defined as follows:

$$\operatorname{dist}_h(Y_1, Y_2) = \frac{1}{m} \Big| \{ \beta = 1, \dots, m \mid y_1^\beta \neq y_2^\beta \} \Big| = \frac{1}{m} \sum_{\beta=1}^{m} \operatorname{diff}(y_1^\beta, y_2^\beta),$$

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_31

where $\operatorname{diff}(t_1,t_2)=\begin{cases}1 & \text{if } t_1\neq t_2,\\ 0 & \text{if } t_1=t_2.\end{cases}$

Introduce the weights vector: $\mathbf{W}=(W,U)=(w_1,\dots,w_n,u_1,\dots u_m)$, where $w_\alpha>0$, $u_\beta>0$ for $\alpha=1,\dots,n$, $\beta=1,\dots,m$, and assume that classes are formed with respect to the *weighted distance*:

$$\begin{aligned}\operatorname{dist}^2_{\mathbf{W}}(\mathbf{X_1},\mathbf{X_2})&=\operatorname{dist}^2_{W,e}(X_1,X_2)+\operatorname{dist}^2_{U,h}(Y_1,Y_2)\\&=\sum_{\alpha=1}^{n}w_\alpha^2(x_1^\alpha-x_2^\alpha)^2+\left(\sum_{\beta=1}^{m}u_\beta\operatorname{diff}(y_1^\beta,y_2^\beta)\right)^2.\end{aligned}\tag{1}$$

To determine the weights vector $\mathbf{W}$ we minimize the total intra-class squared distance:

$$H(\mathbf{W})=\frac{1}{M^2}\sum_{k=1}^{c}\left(\sum_{\mathbf{X}_i,\mathbf{X}_j\in C_k}\operatorname{dist}^2_{\mathbf{W}}(\mathbf{X}_i,\mathbf{X}_j)\right).$$

The distance (1) and hence the objective function $H(\mathbf{W})$ are homogeneous with respect to $\mathbf{W}$. So, to workout an effective minimizing procedure, we assume that the generalized average of the weights is constant:

$$\left(\frac{1}{n+m}\left(\sum_{\alpha=1}^{n}w_\alpha^r+\sum_{\beta=1}^{m}u_\beta^r\right)\right)^{\frac{1}{r}}=1,\quad r\in(0,1).$$

Finally, we get the following constrained minimization problem:

$$\begin{cases}H(W,U)=\dfrac{1}{M^2}\displaystyle\sum_{k=1}^{c}\sum_{i,j\in C_k}\operatorname{dist}^2_{W,U}\big((X_i,Y_i),(X_j,Y_j)\big)\to\min,\\ \displaystyle\sum_{\alpha=1}^{n}w_\alpha^r+\sum_{\beta=1}^{m}u_\beta^r=n+m.\end{cases}$$

We will write $i,j\in C_k$ instead of $\mathbf{X}_i,\mathbf{X}_j\in C_k$ for the sake of simplicity.

To solve this problem we use the method of Lagrange multipliers. The correspondent Lagrange function is as follows:

$$\begin{aligned}&\mathcal{L}(\mathbf{W},\lambda)\\&=\frac{1}{M^2}\sum_{k=1}^{c}\sum_{i,j\in C_k}\operatorname{dist}^2_{W,U}\big((X_i,Y_i),(X_j,Y_j)\big)-\lambda\left(\sum_{\alpha=1}^{n}w_\alpha^r+\sum_{\beta=1}^{m}u_\beta^r-(m+n)\right)\\&=\frac{1}{M^2}\sum_{\alpha=1}^{n}w_\alpha^2\sum_{k=1}^{c}\sum_{i,j\in C_k}(x_\alpha^i-x_\alpha^j)^2+\frac{1}{M^2}\sum_{k=1}^{c}\sum_{i,j\in C_k}\left(\sum_{\beta=1}^{m}u_\beta\operatorname{diff}\big(y_\beta^i,y_\beta^j\big)\right)^2\\&\quad-\lambda\left(\sum_{\alpha=1}^{n}w_\alpha^r+\sum_{\beta=1}^{m}u_\beta^r-(m+n)\right).\end{aligned}$$

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_31

The optimal weight vector is determined from the following conditions:

$$\frac{\partial \mathcal{L}_p(\mathbf{W},\lambda)}{\partial w^\alpha} = \frac{\partial \mathcal{L}_p(\mathbf{W},\lambda)}{\partial u^\beta} = \frac{\partial \mathcal{L}_p(\mathbf{W},\lambda)}{\partial \lambda} = 0, \tag{2}$$

where $\alpha = 1,\dots,n$, $\beta = 1,\dots,m$.

The first equation of (2) yields:

$$\frac{2}{M^2} w_\alpha \sum_{k=1}^{c} \sum_{i,j\in C_k} (x_\alpha^i - x_\alpha^j)^2 = \lambda r w_\alpha^{r-1},$$

therefore

$$w_\alpha = \Lambda_r s_\alpha, \tag{3}$$

where

$$\Lambda_r = \left(\frac{\lambda r}{2}\right)^{-\frac{1}{2-r}}, \tag{4}$$

$$s_\alpha = \left(\frac{1}{M^2} \sum_{k=1^c} \sum_{i,j\in C_k} \left(x_\alpha^i - x_\alpha^j\right)^2\right)^{-\frac{1}{2-r}}. \tag{5}$$

In a similar way, the second equation of (2) implies

$$u_\beta = \Lambda_r z_\beta, \tag{6}$$

where Λ_r is defined in (4), and z_β satisfies the following equation $z_\beta^{r-1} = \sum_{\gamma=1}^m A_{\beta\gamma} z_\gamma$, where

$$A_{\beta\gamma} = \frac{1}{M^2} \sum_{k=1}^{c} \sum_{i,j\in C_k} \operatorname{diff}(y_\beta^i, y_\beta^j) \operatorname{diff}(y_\gamma^i, y_\gamma^j). \tag{7}$$

The third equation of (2) is $\sum_{\alpha=1}^n w_\alpha^r + \sum_{\beta=1}^m u_\beta^r = n+m$. Substituting into it (3) and (6) and eliminating Λ_r, one obtains

$$\Lambda_r = \left(\frac{\sum_{\alpha=1}^n s_\alpha^r + \sum_{\beta=1}^m z_\beta^r}{n+m}\right)^{-\frac{1}{r}} \tag{8}$$

Following [4] we propose a relaxation iterative method for z_β:

$$z_{\beta,\text{next}} = z_\beta - \tau \left(z_\beta^{r-1} - \sum_{\gamma=1}^m A_{\beta\gamma} z_\gamma\right), \tag{9}$$

here τ is a relaxation parameter.

The above considerations are summarized in the algorithm 3.1.

Remark 1. One can show that for small r every feature has a similar contribution to the total intra-class squared distance. For greater r contribution of each feature is proportional to the value of $s_\alpha(r)$ for continuous features and $z_\beta(r)$ for nominal ones.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_31

Algorithm 3.1 Determining of the metric weights

Require: $\mathbb{X} = \{ \mathbf{X}_1, \ldots, \mathbf{X}_M \}$ is the set of records, $C_1, \ldots, C_c$ — the set of classes
Ensure: $\mathbf{W} = (W, U)$ is the optimal weights vector
  Compute s_α, $\alpha = 1, \ldots, n$ with (5)
  Compute matrix A with (7)
  Choose initial z vector as $\mathbf{z} = (1, \ldots, 1)$
  while $\|z_{\text{next}} - z\| > \varepsilon$ **do**
    Compute z_{next} with (9)
  end while
  Compute Λ_r with (8)
  Compute w_α with (3) for $\alpha = 1, \ldots, n$
  Compute u_β with (6) for $\beta = 1, \ldots, m$
  return (W, U)

4 Numerical Experiments

To illustrate the concept a few utilities in C, R and Perl have been created. The code is available as a project on Gitlab at `https://gitlab.com/adenisiuk/weightedhamming`.

We performed tests for values $r = 0.05$, 0.15, 0.35, 0.55, 0.75, and 0.95.

We consider 2 datasets from the UCI Machine Learning Repository [1]: The Australian Credit Approval and The Heart Disease. We tested algorithm on artificial dataset as well.

All the tested datasets were splitted into train (80%) and test (20%) parts.

The purpose of our test is to show that the algorithm improves the standard KNN classification with non-weighted metric. However, we performed also comparison with two powerful classifiers: random forest and SVM. Implementations of these algorithms in R were used in experiments: [10, 11].

The continuous data were normalized before testing the KNN algorithm with non-weighted metric and the SVM classifier.

To compare performance of algorithms we used the area under the ROC curve (AUC) as a measure. It is known that AUC is a suitable measure of binary data classification efficiency [5]. To calculate the AUC we used the R implementation [13].

4.1 Australian Credit Approval

The Australian Credit Approval dataset has 6 continuous, 8 nominal attributes, 690 records, and 2 decision categories. The results of experiments are presented in the table 1. One can see that our algorithm overperforms the standard KNN and its performance is comparable to the random forest and SVM.

Two continuous features in this dataset have values that are bigger than others. The corresponding weights computed by our algorithm are very small and recompense big variance of these two features.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_31

Table 1. Numerical experiments for the *Australian Credit Approval* dataset.

Algorithm	**AUC**	$H(r)$
Weighted KNN, $r = 0.05$	0.942	166.53
Weighted KNN, $r = 0.15$	0.942	103.93
Weighted KNN, $r = 0.35$	0.938	52.12
Weighted KNN, $r = 0.55$	0.947	31.98
Weighted KNN, $r = 0.75$	0.947	21.15
Weighted KNN, $r = 0.95$	0.942	13.66
Unweighted KNN, normalized data	0.925	
Random forest	0.949	
Support Vector Machine	0.941	

Table 2. The results of numerical experiments for the *Heart Disease* data set.

Algorithm	**AUC**	$H(r)$
Weighted KNN, $r = 0.05$	0.966	67.56
Weighted KNN, $r = 0.15$	0.969	54.62
Weighted KNN, $r = 0.35$	0.966	36.41
Weighted KNN, $r = 0.55$	0.974	25.68
Weighted KNN, $r = 0.75$	0.979	19.21
Weighted KNN, $r = 0.95$	0.946	14.55
Unweighted KNN, normalized data	0.953	
Random forest	0.941	
Support Vector Machine	0.966	

4.2 Heart Disease

The Heart Disease dataset has 6 continuous, 7 nominal attributes, 370 records, and 2 decision categories. The results are presented in the table 4.2. We can see that for most values of r our algorithm overperforms the standard KNN. For $r = 0.66$, $r = 0.55$, and $r = 0.75$ our algorithm overperforms all the tested classifiers. Contrary to the previous dataset, we see that the maximum performance corresponds to middle values of r.

4.3 Artificial Dataset

We have tested our algorithm on artificial dataset as well. The artificial dataset was constructed as follows. Each record has 6 continuous and 6 nominal features. The cardinalities of nominal domains were arbitrary chosen as 2, 12, 3, 13, 4, 14. We use the weighted metric (1) with the following weights: 2500, 1010, 1.5, 5.1, 0.001, 0.0001 for the continuous part and the same for the nominal part. Two classes are two balls with random centers in this metric space. The radius of each ball is equal to $1/\sqrt{1.75}$ of distance between the centers. Analyzed data, 1000 for each class were randomly chosen from corresponding balls.

The table 4.2 contains the average rate for 10 generated datasets.

One can see that our algorithm overperforms the standard KNN by 1% and reaches almost 100% performance.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_31

Table 3. The results of numerical experiments for the *Artificial* data set.

Algorithm	**AUC**	$H(r)$
Weighted KNN, $r = 0.05$	0.997	10213.74
Weighted KNN, $r = 0.15$	0.997	1043.18
Weighted KNN, $r = 0.35$	0.997	127.58
Weighted KNN, $r = 0.55$	0.996	49.96
Weighted KNN, $r = 0.75$	0.995	26.68
Weighted KNN, $r = 0.95$	0.995	13.96
Unweighted KNN, normalized data	0.990	
Random forest	0.999	
Support Vector Machine, normalized data	0.990	

Let us however make a remark concerning the weights of the metric (1). Our algorithm discovers stably the continuous part. That means that discovered weights are proportional to the initial ones. But the nominal weights do not reflect initial weights. There can be two reasons of this phenomen. The first one: our way of modelling impact of nominal weights is not reliable. The second one, that is seemed to be more credible: the Hamming distance is too weak to reflect relations between the nominal data.

5 Conclusion and Future Work

In this article we consider modeling of nominal-continuous data. We used the weighted Euclidean metric on the continuous part and the weighted Hamming metric on the nominal one. A new method for automatic weights detection based on minimizing the total inner-class squared distance is proposed. The detected metric then was used in KNN classification.

Numerical experiments on real and artificial data show that our approach can improve the standard KNN classification algorithm and achieve the AUC performance of such sophisticated classification algorithms as random forest or support vector machine classifier, preserving simplicity of the standard KNN.

Summarizing: the proposed method is an interesting proposition for the classification problem. Moreover, as it was mentioned at the end of section 4, it could get further improvement. Specifically, we plan to consider alternatives to the standard Hamming metric for the nominal part.

Besides, the discovered metric can be used in other algorithms for analysis of nominal-continuous data that are based on similarity.

These issues will be explored and presented in the future.

References

1. Asuncion, A., Newman, D.J.: UCI machine learning repository (2007), `http://www.ics.uci.edu/~mlearn/MLRepository.html`

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_31

2. Bellet, A., Habrard, A., Sebban, M.: Metric learning. Springer Cham (2015). https://doi.org/10.1007/978-3-031-01572-4
3. Davis, J.V., Kulis, B., Jain, P., Sra, S., Dhillon, I.S.: Information-theoretic metric learning. In: Proceedings of the 24th International Conference on Machine Learning. p. 209–216. ICML '07, Association for Computing Machinery, New York, NY, USA (2007). https://doi.org/10.1145/1273496.1273523
4. Denisiuk, A., Grabowski, M.: Embedding of the hamming space into a sphere with weighted quadrance metric and c-means clustering of nominal-continuous data. Intelligent Data Analysis **22**(6), 1297001314 (2018). https://doi.org/10.3233/IDA-173645
5. Fawcett, T.: An introduction to roc analysis. Pattern Recognition Letters **27**(8), 861–874 (2006), rOC Analysis in Pattern Recognition
6. Goldberger, J., Hinton, G.E., Roweis, S., Salakhutdinov, R.R.: Neighbourhood components analysis. In: Saul, L., Weiss, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems. vol. 17, pp. 513–520. MIT Press (2004)
7. Karayiannis, N.B., Randolph-Gips, M.M.: Non-euclidean c-means clustering algorithms. Intell. Data Anal. **7**(5), 405–425 (2003)
8. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis, Wiley Series in Probability and Statistics, vol. 344. John Wiley (2008). https://doi.org/10.1002/9780470316801
9. Kulis, B.: Metric learning: A survey. Foundations and Trends® in Machine Learning **5**(4), 287–364 (2013). https://doi.org/10.1561/2200000019
10. Liaw, A., Wiener, M.: Classification and regression by randomforest. R News **2**(3), 18–22 (2002)
11. Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., Leisch, F.: e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien (2022), r package version 1.7-12
12. Norouzi, M., Fleet, D.J., Salakhutdinov, R.R.: Hamming distance metric learning. In: Pereira, F., Burges, C., Bottou, L., Weinberger, K. (eds.) Advances in Neural Information Processing Systems. vol. 25, pp. 1061–1069. Curran Associates, Inc. (2012)
13. Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J.C., Müller, M.: proc: an open-source package for r and s+ to analyze and compare roc curves. BMC Bioinformatics **12**, 77 (2011)
14. Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. In: Thrun, S., Saul, L., Schölkopf, B. (eds.) Advances in Neural Information Processing Systems. vol. 16, pp. 41–48. MIT Press (2003)
15. Shi, Y., Bellet, A., Sha, F.: Sparse compositional metric learning. Proceedings of the AAAI Conference on Artificial Intelligence **28**(1) (Jun 2014). https://doi.org/10.1609/aaai.v28i1.8968
16. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. Journal of Machine Learning Research **10**(9), 207–244 (2009)
17. Xing, E., Jordan, M., Russell, S.J., Ng, A.: Distance metric learning with application to clustering with side-information. In: Becker, S., Thrun, S., Obermayer, K. (eds.) Advances in Neural Information Processing Systems. vol. 15, pp. 521–528. MIT Press (2002)
18. Zhai, D., Liu, X., Chang, H., Zhen, Y., Chen, X., Guo, M., Gao, W.: Parametric local multiview hamming distance metric learning. Pattern Recognition **75**, 250–262 (2018). https://doi.org/10.1016/j.patcog.2017.06.018, distance Metric Learning for Pattern Recognition

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_31