Metric Learning for Nearest Neighbor Classification

Akarin Phaibulpanich*

Department of Statistics, Faculty of Commerce and Accountancy, Chulalongkorn University, Bangkok, Thailand akarin@cbs.chula.ac.th

Kerby Shedden

Department of Statistics, University of Michigan, Ann Arbor, USA
kshedden@umich.edu

We develop methods for constructing an A-weighted metric $(x-y)'A(x-y)$ that improves the performance of K-nearest neighbor (KNN) classifiers. KNN is known to be highly flexible, but can be somewhat inefficient and unstable. By incorporating a parametrically optimized metric into KNN, global dimension reduction is carried out efficiently, leaving the most difficult nonlinear features of the problem to be solved on a low dimensional projected feature space. Optimization over A is done by formulating a probability model that captures KNN's essential property – using only a local neighborhood of training cases to predict the class of a test case. The expected correct vote margin can be calculated under the probability model and optimized over A using gradient methods to yield a metric that is adapted to a particular problem. This framework incorporates variable selection as well as variate selection, in which certain linear combinations of the variables are deemed either informative or completely uninformative. The estimated A matrix can be used for both classification and data analysis, as it contains information about which features are informative (either in a linear or nonlinear sense), or completely uninformative about class membership.

Key words: *A*-weighted metric learning, global dimension reduction, *K*-Nearest Neighbor Classification

1. Introduction

The similarity measure plays an important role in successful *K*-nearest neighbor (KNN) classification, which classifies a test case according the predominant class among its *K* nearest neighbors. In large, complicated data sets, such as those in the fields of computer vision and bioinformatics, it is likely that a few features are much more informative than the rest. In this setting, applying KNN with all features weighted uniformly may yield inferior results. Moreover, it can happen that certain linear combinations of features are especially informative, or completely uninformative. To adapt to these situations, a similarity measure can be identified that is optimized to the characteristics of a particular classification problem.

In this paper, we focus on learning the metric for KNN of the form

$$D(x,y) = (x-y)'A(x-y), \tag{1.1}$$

Where x, y are two data points in d dimensions and A is a $d \times d$ matrix. Classification based on $D(\cdot,\cdot)$ will be called "A-weighted KNN". The matrix A must be positive semi-definite to ensure the metric properties (non-negativity and triangle inequality).

In many classification problems, information used to determine the class label is encoded in fewer than d projected variates. In our first example, we show that the existence of uninformative features can diminish KNN performance.

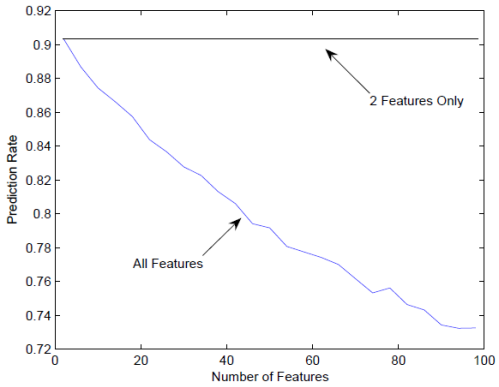

Figure 1.1: Weighing is beneficial when not all features are equally informative. These are graphs of KNN (K=10) correct classification rates as functions of the number of features. The results are shown based on all features, and based only on the two informative features.

Here, two classes are generated from multivariate normal distributions where only the first two of the d features are informative in distinguishing the two classes. Specifically, we let $x_i \sim N_d(\boldsymbol{\mu}_i, \Sigma), i = 1,2,$ where $\boldsymbol{\mu}_i = \mathbf{0}$, $\boldsymbol{\mu}_2(1) = \boldsymbol{\mu}_2(2) = 2$, $\boldsymbol{\mu}_2(3) =, \dots, \boldsymbol{\mu}_2(d) = 0$ and $\Sigma = I_{d\times d}$. Figure 1.1 presents two plots of the KNN (K=10) correct classification rates, using Euclidean distance to define the neighborhoods, plotted as functions of the number of features (d). Evidently, the more uninformative features present in the data, the worse the prediction rates become. This evidence supports the use of feature weighting in high-dimensional problems in which not all features are equally informative.

2. Metric Learning via Model-based KNN

We consider 2-class classification problem: let $x_{11}, \dots, x_{n_1 1} \in \Re^d$ and $x_{12}, \dots, x_{n_2 2} \in \Re^d$ be training observations from class C_1 and class C_2 respectively, and let the total training sample be of size$n = n_1 + n_2$. Our goal is to learn a similarity measure based on the A-weighted squared Euclidean distance (1.1), by identifying a positive semidefinite matrix A that performs well for a type of nearest neighbor (KNN) prediction.

A

The empirical misclassification rate for KNN, using neighborhoods based on $D(\cdot,\cdot;A)$ is not easy to optimize with respect to A because it is a non-smooth function. To make the optimization tractable, we consider a model-based formulation of KNN, where the model-based estimate of the misclassification rate becomes a sooth function of A.

For a give test case t and a fixed similarity measure $s(\cdot,\cdot)$, let $\pi_1(\cdot;t)$ be the distribution of similarities between t and a case randomly selected from class C_1. Similarly, let $\pi_2(\cdot;t)$ be the distribution of similarities between t and a case randomly selected from class C_1. Suppose that the proportion of class C_1 in the population is p, and let $q(t)$ be the 95^{th} percentile of the mixture distribution

$$\pi(\cdot;t) = p\pi_1(\cdot;t) + (1-p)\pi_2(\cdot;t). \qquad (2.1)$$

For a given training set of size n, if we select the $0.05n$ most similar cases in the training set to the test case, it is expected that

$$Q_1(t) = np \int_{q(t)}^{\infty} \pi_1(z;t)\, dz \tag{2.2}$$

of the selected cases will be from class C_1 and

$$Q_2(t) = n(1-p) \int_{q(t)}^{\infty} \pi_2(z;t)\, dz \tag{2.3}$$

will be from class C_2.

In the KNN algorithm, the predicted class for test case t is the class that is most represented among the K most similar training cases. Therefore in our model for KNN, class C_1 is predicted if $Q_1(t) > Q_2(t)$, and class C_2 is predicted otherwise.

Let $\pi_1(\cdot;t,A)$ and $\pi_2(\cdot;t,A)$ be the densities denoted $\pi_1(\cdot;t)$ and $\pi_2(\cdot;t)$ above, for the similarity $s = -\log D(\cdot,\cdot;A)$ determined by a given value of A. Let $\hat{\pi}_1(\cdot;t,A)$ and $\hat{\pi}_2(\cdot;t,A)$ be the estimated distributions based on training data. Let $q(t;A)$ be the 95^{th} percentile of the mixture distribution

$$\hat{\pi}(\cdot;t,A) = p\hat{\pi}_1(\cdot;t,A) + (1-p)\hat{\pi}_2(\cdot;t,A), \tag{2.4}$$

and let

$$Q_1(t,A) = np \int_{q(t,A)}^{\infty} \hat{\pi}_1(z;t,A)\, dz \tag{2.5}$$

$$Q_2(t,A) = n(1-p) \int_{q(t,A)}^{\infty} \hat{\pi}_2(z;t,A)\, dz \tag{2.6}$$

Therefore, the number of correct classifications on the test data set can be expressed as the following sum of indicator functions:

$$\sum_{t\in C_1} \mathrm{I}[Q_1(t,A) > Q_2(t,A)] + \sum_{t\in C_2} \mathrm{I}[Q_1(t,A) < Q_2(t,A)] \tag{2.7}$$

where it is understood that $Q_1(t,A)$ and $Q_2(t,A)$ are calculated without using t.

The value of A that optimizes (2.8) is difficult to calculate due to the indicator functions. To create a tractable objective function similar to (2.7), but also continuous, we will relax the indictor functions with a computable and unbiased estimate of the correct prediction rate using cross-validation on the training set

$$\sum_{t\in C_1} [Q_1(t,A) - Q_2(t,A)] + \sum_{t\in C_2} [Q_2(t,A) - Q_1(t,A)] \tag{2.9}$$

Note that for a given t, $Q_1(t,A) - Q_2(t,A)$ and $Q_2(t,A) - Q_1(t,A)$ can be interpreted as the estimated expected vote margin for classes C_1 and C_2, respectively. Our goal will be to optimize the vote margin with respect to A, which takes the following form:

$$\frac{1}{n}\left(\sum_{k=1}^{n_1} [Q_1(x_{k1},A) - Q_2(x_{k1},A)] + \sum_{k=1}^{n_2} [Q_2(x_{k2},A) - Q_1(x_{k2},A)]\right) \tag{2.10}$$

Since we use $s(x,z;A) = -\log D(x,z;A)$ as a similarity measure, $-\log()$ transforms distribution of $D(x,z;A)$ into a more normal shape to some extent. Thus, we

explicitly express the correct vote margin objective function in terms of the normal models for the $\pi_k(\cdot;t)$ distributions. The normal model also incorporates both means and variances which are needed to prevent the classification from reducing to the comparison of the means, while the percentile of a normal distribution is easy to calculate.

For each k in the first summation in (2.10):

Let

$$\pi_1(z;x_{k1},A)\sim N(\mu_1(x_{k1},A),\sigma_1^2(x_{k1},A))$$
$$\pi_2(z;x_{k1},A)\sim N(\mu_2(x_{k1},A),\sigma_2^2(x_{k1},A))$$

These normal parameters, $\mu_1(x_{k1},A),\mu_2(x_{k1},A),\sigma_1^2(x_{k1},A),\sigma_2^2(x_{k1},A)$, will be estimated through the usual sample mean and variance formula:

$$\hat{\mu}_1(x_{k1},A) = -\frac{1}{n_1}\sum_{j\neq k}\log\big((x_{k1}-x_{j1})A'(x_{k1}-x_{j1})\big)$$
$$\hat{\mu}_2(x_{k1},A) = -\frac{1}{n_2}\sum_{j}\log\big((x_{k1}-x_{j2})A'(x_{k1}-x_{j2})\big)$$
$$\hat{\sigma}_1^2(x_{k1},A) = \frac{1}{n_1-2}\sum_{j\neq k}[-\log\big((x_{k1}-x_{j1})A'(x_{k1}-x_{j1})\big)-\hat{\mu}_1(x_{k1},A)]^2$$
$$\hat{\sigma}_2^2(x_{k1},A) = \frac{1}{n_2-1}\sum_{j}[-\log\big((x_{k1}-x_{j2})A'(x_{k1}-x_{j2})\big)-\hat{\mu}_2(x_{k1},A)]^2$$

Let $q(x_k,A)$ be the 95^{th} percentile of the mixture

$$\pi(z;x_k,t) = p\pi_1(x_k;t)+(1-p)\pi_2(x_k;t),$$

which is the solution of the equation

$$0.05 = p\left(1-\Phi\left(\frac{q_1(x_k,A)-\hat{\mu}_1(x_{k1},A)}{\hat{\sigma}_1(x_{k1},A)}\right)\right)+(1-p)\left(1-\Phi\left(\frac{q_1(x_k,A)-\hat{\mu}_2(x_{k1},A)}{\hat{\sigma}_2(x_{k1},A)}\right)\right),$$

where $\Phi(\cdot)$ represents the standard normal cdf. Note that this equation can be sovled through standard numerical methods such as bisection or Newton-Raphson methods.

Thus, we calculate $Q_1(x_{k1},A)$ as

$$Q_1(x_{k1},A) = np\int_{q(x_{k1},A)}^{\infty}\pi_1(z;x_{k1},A)\,dz = np\left(1-\Phi\left(\frac{q_1(x_k,A)-\hat{\mu}_1(x_{k1},A)}{\hat{\sigma}_1(x_{k1},A)}\right)\right).$$

And $Q_2(x_{k1},A)$ and the quantities in the second summation of (2.10) can be obtained similarly.

Therefore, under the normal models for $\pi_k(\cdot;t)$, the model-based expression for the KNN correct prediction rate can be written as

$$G(A) = \frac{1}{n}\left(\sum_{k=1}^{n_1}[Q_1(x_{k1},A)-Q_2(x_{k1},A)]+\sum_{k=1}^{n_2}[Q_2(x_{k2},A)-Q_1(x_{k2},A)]\right)$$

Our goal is now to find the matrix A that optimizes the nonlinear function $G(A)$, with the constraint that A must be a positive semi-definite matrix. This constraint can actually be avoided by optimizing $G(A)$ with respect to $A^{1/2}$ rather than A. Doing so will cause the number of parameters to increase from $d(d-1)/2$ to d^2, resulting in a more expensive computation. However, from our experience, the search for the optimal $A^{1/2}$for a problem of $d=10$, with $n_1,n_2\leq 100$ typically takes only a few hours.

The optimization can be done either with or without gradient information. Without gradient and without the positive semi-definite constraint, we can use either Nelder-Mead simplex method, Quasi-Newton method with a mixed quadratic and cubic line search procedure, or sequential quadratic programming method as the searching tools. We use $A = I$ as the starting value.

3. Results

3.1 Noisy XOR data

Adapted from Lowe (1995), the XOR data is an example where the classification is determined by the interaction of two features, with some noise features also present. We let the number of features d be 4. The features are generated from mixtures of two normal distributions. Each sample is generated by first generating δ_1 and δ_2 as 0 or 1 with equal probability that is $P(\delta_1 = 1) = P(\delta_1 = 0) = 1/2$ and $P(\delta_2 = 1) = P(\delta_2 = 0) = 1/2$ Then, we generate the feature values as follows:

$$x_1 = \delta_1 N(0,0.09) + (1-\delta_1)N(1,0.09)$$
$$x_2 = \delta_2 N(0,0.09) + (1-\delta_2)N(1,0.09)$$
$$x_3 = \delta_1 N(0,0.25) + (1-\delta_1)N(1,0.25)$$
$$x_4 = \delta_2 N(0,0.25) + (1-\delta_2)N(1,0.25)$$

The class labels are assigned by the XOR function: $\delta_1\delta_2 + (1-\delta_1)(1-\delta_2)$.

Since the third and the fourth features are redundant and noisier than the first two, and both features 1 and 2 are equally informative in class determination, the optimal A would be

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

We generated two hundred training and two hundred test samples as described above. Our estimated A is

$$\hat{A} = \begin{pmatrix} 3.2119 & -0.0691 & 0.5754 & -0.0143 \\ -0.0691 & 1.8179 & 0.0598 & 0.5807 \\ 0.5754 & 0.0598 & 0.1059 & 0.0205 \\ -0.0143 & 0.5807 & 0.0205 & 0.1855 \end{pmatrix}$$

From the prediction results on the 200 test samples, we find that A-weighted metric does improve the classification rates for KNN from 0.8 to 0.85. Because of the interaction between features, LDA performs very poor in this case (prediction rate equals 0.52), while QDA and SVM with radial basis have no trouble classifying the XOR data (prediction rates are 0.9 and 0.9, respectively).

3.2 Logistic Regression

In this logistic regression example, feature values are generated from the standard multivariate normal distribution, $\boldsymbol{x} = N_6(\boldsymbol{0}, \boldsymbol{I})$, with class assignments determined stochastically by a linear function of the features as

$$\log\left(\frac{P(Y=2|\boldsymbol{x})}{P(Y=1|\boldsymbol{x})}\right) = \theta'\boldsymbol{x},$$

And we let $\theta(i) = 2/\sqrt{6}$ for $i \leq 3$ and $\theta(i) = -2/\sqrt{6}$ for $i > 3$. As a result, the class information only lies in the subspace spanned by θ.

The estimated A has eigenvalues (13.25, 0.001, 0.0001, 0, 0, 0)', asserting that KNN classification is best done in a one-dimensional space. With this estimated A, A-weighted metric improves KNN classification rate from 0.68 to 0.80, which is comparable to the prediction rate from SVM.

3.4 Pima data

The Pima data from the National Institute of Diabetes and Digestive and Kidney Disease contains diabetes information on 768 females of Pima Indian heritage who are at least 21 years old. There are 8 numerical features and the response variable is a positive diabetes test. We randomly select 200 observations as training data sets, and we randomly select 200 samples from the remaining data as test data. Model-based KNN achieves a slightly better classification at 0.76 compared to 0.70 from KNN. However the method is inferior to LDA and SVM which produces the classification rates at 0.78 and 0.80 respectively.

4. Conclusion

In this paper, a new metric learning for KNN is developed. A classifier analogous to KNN, called the model-based KNN, is first introduced. This method classifies a test case by comparing the expected vote margin under normal models rather than comparing the number of training samples from each class among the K top votes. Using normal models allows us to obtain a smooth objective function, in which the optimal matrix A can be found via standard optimizers. We have tested our metric learning approach on simulated and real data. In each of the simulated examples, the method is able to identify the true class dimension and the linear combinations between features that are informative. Using the learned A-weighted metric, the model-based KNN have shown to perform better than KNN. We have also learned that the results for KNN using A-weighted metric tends to be between those of KNN and SVM.

5. References

Aha, D.W.(1992) "Tolerating noisy, irrelevant, and novel attributes in instance-based learning algorithm," *International Journal of Man-Machine Studies*, 36:267-287.

Bache, K. & Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Broyden, C.G. (1970), "The convergence of a class of double-rank minimization algorithm," *Journal of Institute of Mathematics Application*, 6:76-90.

Coleman, T.F., Li, Y., (1992) "On the convergence of reflective newton methods for large-scale nonlinear minimization subject to bounds," *Mathematical Programming*, 67:189-224.

Coleman, T.F., Li, Y., (1996) "An interior, trust region approach for nonlinear minimization subject to bounds," *SIAM Journal on optimization*, 6:418-445.

Shanno, D.F. (1970) "Conditioning of quasi-newton methods for function minimization," *Mathematics of computing*, 24:647-656.

Han, S.P. (1975) "A globally convergent method for nonlinear programming," *Journal of optimization Theory and Applications*, 22:297.

Hastie, T., Tibshirani, R., and Friedman, J. (2001) "The Element of Statistical Learning," Springer, first edition.

Lowe, D.G., (1995) "Similarity metric learning for a variable-kernel classifier," *Neural Computation*, 7:72-85.

Nelder, J.A., Mead, R. (1965) "A simplex method for function minimization," *Computing*, 7:308-313.

Powell, M.J.D. (1991) "A fast algorithm for nonlinearly constrained optimization calculations" *Numerical Analysis*, 630:297