5

Similarity Discriminant Analysis

Luca Cazzanti
Applied Physics Lab
Box 355640
University of Washington, Seattle, WA 98105,
USA

1. Introduction

This chapter details *similarity discriminant analysis* (SDA), a new framework for similarity-based classification. The two defining characteristics of the SDA classifica- tion framework are *similarity-based* and *generative*. The classifiers in this framework are similarity-based, because they classify based on the pairwise similarities of data samples, and they are generative, because they build class-dependent probability models of the similarities between samples. Similarity-based classifiers already exist; classifiers based on generative models already exist. SDA is a *new* framework for classification comprising classifiers that are *both* similarity-based and generative.

Within the general SDA framework, this chapter describes several families of classifiers: the *SDA classifier*, the *local SDA classifier*, and the *mixture SDA classifier*. The SDA classifier is at the foundation of SDA. It classifies based on the class-conditional generative models of the similarity of the samples to representative class prototypes, or *centroids*. The SDA framework is introduced, developed, and discussed with the aid of this centroid-based SDA classifier. Then, the centroid-based SDA classifier is generalized beyond class centroids to arbitrary class-descriptive statistics. Other possible statistics are described, illustrating the power and generality of the SDA framework.

The local SDA classifier is a local version of the SDA classifier. It builds similarity-based class-conditional generative models within a neighborhood of a test sample to be classified. The local class models are endowed with low bias and retain the powerful quality of interpretability associated with generative probability models. Local SDA is a consistent classifier, in the sense that its error rate converges to the Bayes error rate, which is the best possible error rate attainable by a classifier.

The mixture SDA classifier draws from the well-established metric learning mixture model research. It generalizes the single-centroid SDA classifier to a mixture of single-centroid SDA components. The mixture SDA classifier can be trained with an expectation-maximization (EM) algorithm which parallels the standard EM approach for the well-known Gaussian mixture models.

The problem of classifying samples based only on their pairwise similarities may be divided into two sub-problems: measuring the similarity between samples and classifying the samples based on their pairwise similarities. It is beyond the scope of this chapter to discuss exhaustively and in detail various ways to measure similarity and various similarity-based

Source: Machine Learning, Book edited by: Abdelhamid Mellouk and Abdennacer Chebira,
ISBN 978-3-902613-56-1, pp. 450, February 2009, I-Tech, Vienna, Austria

classifiers. The reader is referred to the references for more details; here, only a brief summary of relevant techniques is provided

1.1 Measuring similarity

Judging similarity between samples characterized by many disparate data types poses challenges of data representation and quantitative comparison. For example, modern databases store information from disparate data sources in different formats: multimedia databases store audio, video and text data; proteomics databases store information on proteins, genetic sequences, and related annotations; internet traffic databases store mouse click histories, user profiles, and marketing rules; homeland security databases may store data on individuals and organizations, annotations from intelligence reports, and maritime shipping records. These database objects, or samples, are described by both numerical and non-numerical data. For example, a security database might store cell phone records in textual form and voice parameters for speaker recognition in numerical form. Representing all these different data types with continuous-valued numbers in a geometric feature space is not appropriate. Thus, current metric space classifiers which rely on metric similarity functions may not be applicable.

Furthermore, in some applications, only the pairwise similarities may be observed, and the underlying features may be inaccessible. For example, one of the datasets discussed in this chapter consists of human-judged similarities between pairs of sonar echoes. For this dataset, the putative perceptual features from which the human similarity ratings are generated are unknown - indeed eliciting the features remains an ongoing research problem (Philips et al., 2006) - but the similarity ratings are nonetheless successfully used for classification. In many applications, the similarity relationship between samples may lack the metric properties usually associated with distance (minimality, symmetry, triangle inequality); thus, using a metric function to express the pairwise similarities is suboptimal. Similarities are more general than distances and require more general functions than metrics (Tversky, 1977). Several researchers have addressed the problem of measuring similarity by rpoposing several simialrity measures. Psychologists, leacd by Tversky, have proposed models of similarity that take into account context and the non-metric way in which humans judge the similarity between complex objects (Tversky, 1977; Tversky & Gati, 1978; Gati & Tversky, 1984; Sattath & Tversky, 1987). The value difference metric (VDM) was originally designed with the goal of improving nearest-neighbor classification (Stanfill & Waltz, 1986) of text documents, and subsequent improvements extended it to classification of objects characterized by both textual and numerical features (Wilson & Martinez, 1997; Cost & Salzberg, 1993). Lin proposed an information-theoretic similarity (Lin, 1998) for document retrieval; (Cazzanti & Gupta, 2006) proposed the *residual entropy similarity* measure by extending Tversky's psychological similarity models with information-theoretic notions, and showed that it strongly takes into account the context in which the similarity is being evaluated. More comprehensive reviews of similarity measures appear in (Santini & Jain, 1999) and (Everitt & Rabe-Hesketh, 1997).

1.2 Similarity-based classifiers

Similarity-based classifiers are defined as those classifiers that require only a pairwise similarity - a description of the samples themselves is not needed. Similarity-based classifiers classify test samples given a labeled set of training samples, the pairwise similarity values of the training samples, and the similarity of the test sample to the training samples. If the description of the samples in terms of feature vectors is available, an existing or ad hoc similarity function that maps any two samples to a similarity value may be used (Bicego et al., 2006; Pekalska et al., 2001; Jacobs et al., 2000; Hochreiter & Obermayer, 2006).

Among the existing similarity-based classifiers, the simplest method is the nearest neighbor classifier, which determines the most similar training sample *z* to the test sample *x*, and classifies *x* as *z*'s class:

$$\hat{y} = \arg \max_{h=1,\ldots,G} \left(\max_{z \in \mathcal{X}_h} s(x,z) \right), \tag{1}$$

where $\mathcal{X}_h$ is the set of training samples from class *h*. More generally, the *k*-nearest neighbor classifier (*k*-NN) determines a neighborhood of *k* most similar training samples to the test sample *x*, and classifies *x* as the most-frequently occurring class label among the neighbors. Experiments have shown that nearest neighbors can perform well on practical similarity-based classification tasks (Cost & Salzberg, 1993; Pekalska et al., 2001; Simard et al., 1993; Belongie et al., 2002). For example, nearest neighbor classifiers using a tangent distortion metric and a shape similarity metric have both been shown to achieve very low error on the MNIST character recognition task.

Condensed near-neighbor strategies replace the set of training samples for each class with a set of prototypes for that class. Usually the prototype set is an edited set of the original training samples (also called edited nearest neighbors), but the prototypes do not need to be from the original training set. Let c_h be the number of the prototypes $\{\mu_{hl}\}$ for class *h*; then, the condensed nearest neighbor rule is to classify a test sample *x* as the class of the prototype to which it is most similar,

$$\hat{y} = \arg \max_{h=1,\ldots,G} \left(\max_{l=1,\ldots,c_h} s(x,\mu_{hl}) \right) \tag{2}$$

Many authors have considered strategies for condensing near-neighbors for similarity-based classification to increase classification speed, decrease the required memory, remove outliers, and possibly attain better performance (Weinshall et al., 1999; Jacobs et al., 2000; Lam et al., 2002; Pekalska et al., 2006; Lozano et al., 2006). A well-known strategy for condensing nearest neighbors in non-metric spaces is the *k-medoids* algorithm (Hastie et al., 2001). Given a set of c_h candidate prototypes selected from $\mathcal{X}_h$, the remaining training samples $z \in \mathcal{X}_h$ are assigned to their nearest (most similar) prototype, so that the set $\mathcal{X}_h$ of all training samples from class *h* is partitioned in c_h mutually-exclusive subsets $\{\mathcal{X}_{hl}\}$, and each $\mathcal{X}_{hl}$ is uniquely associated with candidate prototype μ_{hl}. Then, the *l*th prototype for the *h*th class is updated according to the standard maximum similarity update rule, which selects the new μ_{hl} as the training sample in $\mathcal{X}_{hl}$ which is most similar to all other samples in $\mathcal{X}_{hl}$,

$$\mu_{hl}^{*} = \arg \max_{\mu_{hl} \in \mathcal{X}_{hl}} \sum_{z \in \mathcal{X}_{hl}} s(z,\mu_{hl}). \tag{3}$$

The training samples are then reassigned to the updated prototypes, and the update rule (3) is repeated. The reassignment and update steps are repeated until a predetermined maximum number of iterations is reached or until the updated prototypes $\mu^*_{hl} = \mu_{hl}$ for all h and l. The number of prototypes in each class c_h is determined by cross-validation; the initial prototypes $\{\mu_{hl}\}$ are selected randomly from the training set.

An extreme form of condensed near-neighbors is to replace each class's training samples by one prototypical sample, often called a *centroid*. The resulting nearest centroid classifier can be considered a simple parametric model (Weinshall et al., 1999), though it lacks a probabilistic structure. Let $s(x, z)$ be the similarity between a sample x and a sample z, and let there be a finite set of classes 1, 2, ... ,G. The nearest centroid approach classifies x as the class

$$\hat{y} = \arg \max_{h=1,\dots,G} s(x, \mu_h), \tag{4}$$

where μ_h is the representative centroid for the class h. A standard definition for the centroid of a set of training samples is the training sample that has the maximum total similarity to all the training samples of the same class (Weinshall et al., 1999; Jacobs et al., 2000):

$$\mu_h = \arg \max_{\mu \in \mathcal{X}_h} \sum_{z \in \mathcal{X}_h} s(z, \mu). \tag{5}$$

A variation of the nearest centroid classifier is the local nearest centroid classifier, which is an analog to the local nearest means classifier proposed by Mitani and Hamamoto (Mitani & Hamamoto, 2006, 2000). In this variant, the class centroids (5) are computed from a local neighborhood of each test point x; they are not computed from the entire training set. The neighborhood may be defined in many ways. The most common definition is the k-nearest neighbors. In this case, local nearest centroid is like the k-NN classifier, except that it classifies x as the class of its nearest centroid where the centroids are computed from the k-nearest neighbors of x.

The nearest centroid classifier is analogous to the nearest-mean classifier in Euclidean space, which is the optimal Euclidean-based classifier if one assumes that the class-conditional distributions are Gaussian, the class priors are equal, and that each class covariance is the identity matrix (Duda et al., 2001; Hastie et al., 2001).

2. Similarity discriminant analysis

In standard metric learning, quadratic discriminant analysis (QDA) is a generative classifier that generalizes the nearest-mean classifier by modeling each class-conditional distribution as a Gaussian (Duda et al., 2001). Analogously, SDA is a generative similarity-based classifier that generalizes the nearest-centroid classifier (Weinshall et al., 1999) by modeling each class-conditional distribution with a parametric probability model (Cazzanti et al.; Gupta et al., 2007). The SDA class-conditional probability models have exponential form, because they are derived as the maximum entropy distributions subject to constraints on the mean similarities of the data to the class centroids. As with other parametric approaches to classification, the resulting log-linear SDA classifier is powerful when it effectively models the true generating distribution. This section introduces SDA and shows how it classifies; then, it extends SDA from using class centroids to using arbitrary descriptive statistics to discriminate between the classes, including continuous-valued statistics.

2.1 A generative centroid-based classifier

Assume a class centroid μ_h has been determined for the hth class, where $h = 1, ..., G$. A problem with the nearest centroid classifier given in (4) is that it does not take into account the variability of the similarities to the centroid within a class. To take into account this variability, first consider a simple generalization of nearest centroid, here called the *adjusted nearest centroid classifier* : classify a test sample x as class $\hat{y}$ where

$$\hat{y} = \arg \max_{h=1,\ldots,G} \frac{s(x, \mu_h)}{\bar{s}_{hh}}, \tag{6}$$

and where $\bar{s}_{hh}$ is the average similarity of class h samples to the class h centroid,

$$\bar{s}_{hh} = \frac{1}{n_h} \sum_{z \in \mathcal{X}_h} s(z, \mu_h),$$

where $n_h = |\mathcal{X}_h|$. The adjusted nearest centroid classifier is analogous to the one-dimensional Gaussian rule of classifying based on the the variance-weighted distances to the class means, $\|x- \tilde{\mu}_h\| / \tilde{\sigma}_h$, where $x, \tilde{\mu}_h, \tilde{\sigma}_h \in \mathbb{R}$. The adjusted nearest centroid classifier is more flexible than the nearest centroid classifier, but lacks a probabilistic structure, and takes into account only the similarity of a sample to one class centroid.

Thus, a generative centroid-based classifier that models the probability distribution of the test sample similarity statistics $s(x, \mu_h)$ for each h is proposed. Begin with the Bayes classifier (Hastie et al., 2001), which assigns a test sample x the class $\hat{y}$ that minimizes the expected misclassification cost,

$$\hat{y} = \arg \min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f, g) P(Y = g|x), \tag{7}$$

where $C(f, g)$ is the cost of classifying the test sample x as class f if the true class is g and $P(g|x)$ is the probability that sample x belongs in class g. In practice the distribution $P(g|x)$ is generally unknown, and thus the Bayes classifier of (7) is an unattainable ideal.

Assume that all test and training samples come from some abstract space of samples $\mathcal{B}$, which might be an ill-defined space, such as $\mathcal{B}$ is the set of all amino acids, or $\mathcal{B}$ is the set of all terrorist events, or $\mathcal{B}$ is the set of all women who gave birth to twins. Let $x, \mu_h, z \in \mathcal{B}$, and let the similarity function be some function $s : \mathcal{B} \times \mathcal{B} \rightarrow \Omega$, where $\Omega \subset \mathbb{R}$. If the set of possible samples $\mathcal{B}$ is finite, then the space of the pairwise similarities Ω will also be finite, and hence discrete. For simplicity, in this section assume that Ω is a finite discrete space. Continuous and possibly infinite spaces B, Ω are briefly discussed in Section 2.2.3.

Consider a random test sample X with random class label Y, where x will denote a realization of X. Assume that the relevant information about X's class label is captured by the set $\mathcal{T}(X)$ of G descriptive statistics

$$\mathcal{T}(X) = \{s(X, \mu_1), s(X, \mu_2), \ldots, s(X, \mu_G)\}.$$

That is, the relevant information about x is captured by its similarity to each class centroid. Under this assumption, given a particular test sample x, the classification rule (7) becomes: classify x as class $\hat{y}$ that solves

$$\arg \min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g)P(Y=g|\mathcal{T}(x)).$$

Using Bayes rule, this is equivalent to the problem

$$\arg \min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g)P(\mathcal{T}(x)|Y=g)P(Y=g). \tag{8}$$

Note that $P(\mathcal{T}(x) \mid Y = g)$ is the probability of seeing a particular set of similarities between the test sample x and the G class centroids $\{\mu_1, \mu_2, \ldots, \mu_G\}$ given that x is a class g sample.

Next, assume that each unknown class-conditional distribution $P(\mathcal{T}(x) \mid Y = g)$ has the same average value as the training sample data from class g. That is, given a random test sample X there will be a random similarity $s(X, \mu_h)$; constrain the class-conditional distribution $P(\mathcal{T}(x) \mid Y = g)$ such that

$$E_{P(\mathcal{T}(x)|Y=g)}[s(X,\mu_h)] = \frac{1}{n_g} \sum_{z \in \mathcal{X}_g} s(z,\mu_h), \tag{9}$$

holds for each g and h where n_g is the number of training samples of class g. Each constraint requires that the class-conditional expectation of one of the elements of $\mathcal{T}(X)$ is equal to the maximum likelihood estimate of that element given the training data. This makes for G constraints for each class-conditional distribution, for a total of G×G constraints because there are G class-conditional distributions. Given these constraints, there is some compact and convex feasible set of class-conditional distributions. A feasible solution will always exist because the constraints are based on the data.

As prescribed by Jaynes' principle of maximum entropy (Jaynes, 1982), a unique class-conditional joint distribution is selected by choosing the maximum entropy solution that satisfies (9). Maximum entropy distributions have the maximum possible uncertainty, such that they are as uniform as possible while still satisfying given constraints. Given a set of moment constraints, the maximum entropy solution is known to have exponential form (Cover & Thomas, 1991). For example, in standard metric learning, the Gaussian class-conditional distribution model used in LDA and QDA is the maximum entropy distribution given a specific mean vector and covariance matrix (Cover & Thomas, 1991).

The maximum entropy distribution that satisfies the moment constraints specified in (9) is

$$\hat{P}(\mathcal{T}(x)|Y=g) = \gamma_g e^{\left(\sum_{h=1}^{G} \lambda_{gh} s(x,\mu_h)\right)}, \tag{10}$$

where $\{\gamma_g, \lambda_{g1}, \lambda_{g2}, \ldots, \lambda_{gG}\}$ are a unique set that ensures that the constraints (9) are satisfied and that $\hat{P}$ ($\mathcal{T}$(x) | Y = g) is non-negative and normalized. Rewrite equation (10) as

$$\hat{P}(\mathcal{T}(x)|Y=g) = \prod_{h=1}^{G} \gamma_{gh} e^{\lambda_{gh} s(x,\mu_h)} \tag{11}$$

where $\prod_h \gamma_{gh} = \gamma_g$. Let

$$\hat{P}(s(x,\mu_h)|Y=g) = \gamma_{gh} e^{\lambda_{gh} s(x,\mu_h)};$$

then (11) can be written

$$\hat{P}(\mathcal{T}(x)|Y=g) = \prod_{h=1}^{G} \hat{P}(s(x,\mu_h)|Y=g).$$

That is, under the maximum entropy assumption, the joint distribution on $\mathcal{T}(x)$ is the product of the marginal distributions on each similarity statistic comprising the set $\mathcal{T}(X)$. Thus, the similarity statistics are conditionally independent given the class label under this model. Although one does not expect this conditional independence to be strictly valid, the hypothesis is that it will be an effective model, just as the naive Bayes' model that features are independent is optimistic but useful.

Substituting the maximum entropy solution (10) into (8) yields the classification rule: classify x as the class $\hat{y}$ which solves

$$\arg \min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g) \left(\prod_{h=1}^{G} \gamma_{gh} e^{\lambda_{gh} s(x,\mu_h)} \right) P(Y=g). \tag{12}$$

To solve for the parameters $\{\lambda_{gh}, \gamma_{gh}\}$, one solves the G constraints individually for λ_{gh}. Then given $\{\lambda_{gh}\}$, the $\{\gamma_{gh}\}$ are trivially found using the normalization constraint. Solving for λ_{gh} is straightforward; for example, one uses the Nelder-Mead optimizer built into Matlab (version 15) in the `fminsearch()` function (Mat). This is the method used throughout this work. As an alternative, one may find the probability mass function with maximum entropy, subject to the constraints, without a priori knowledge that the solution is exponential.

The classifier given in (12) is termed the *similarity discriminant analysis* (SDA).

2.2 General generative models for similarity-based classification

The previous section introduced SDA for the case when the descriptive statistics are the similarities of the samples to the class centroids. This section generalizes SDA to arbitrary descriptive statistics $\mathcal{T}(x)$ which can be used to discriminate different classes and describes the resulting general generative model for classifying with arbitrary statistics.

2.2.1 Descriptive statistics

Several possibilities for the descriptive statistics $\mathcal{T}(x)$ are described below.

- Centroid Definitions - A standard centroid definition was given in (5). Another choice is to allow a class prototype that is not constrained to be a training sample,

  $$\mu_h^* = \arg \max_{\mu \in \mathcal{B}} \sum_{z \in \mathcal{X}_h} s(z,\mu). \tag{13}$$

  In this case the solution μ_h^* requires a description of the entire space of possible samples $\mathcal{B}$. In practice, one may not know the entire sample space $\mathcal{B}$, only the training samples $\mathcal{X}$, so it may not be possible to calculate μ_h^*.

A third definition of a class prototype is based on Tversky's analysis of similarity-based near-neighbor relationships (Tversky & Hutchinson, 1986; Schwartz & Tversky, 1980), and takes into account the similarity-based ranks of a training sample's near-neighbors. Define the neighborhood $\mathcal{N}(z) \subseteq \mathcal{X}$ of a sample z as the set of training samples whose nearest neighbor in similarity space is z. The popularity of z is the size of its neighborhood $|\mathcal{N}(z)|$. The class centroid is the sample with the highest popularity, that is,

$$\mu_h = \arg\max_{z \in \mathcal{X}_h} |\mathcal{N}(z)|. \tag{14}$$

This centroid is the training sample that is most often the closest neighbor of the training samples in the class. Ties in popularity are broken by selecting the sample with the highest total similarity to its neighbors.

- Higher Order and Non-Centroidal Descriptive Statistics - Given a set of class centroids $\{\mu_h\}$, higher-order statistics could be used as, or added to, the set of descriptive statistics $\mathcal{T}(X)$, such as $(s(X, \mu_h) - E[s(X, \mu_h)])^2$, or cross-class statistics, such as $(s(X, \mu_h) - E[s(X, \mu_g)])^2$. Or, instead of the centroid-based statistics f$s(X, \mu_h)$g, it might be more appropriate to use the nonparametric statistics formed by the total pairwise similarity for each class h, such that the hth descriptive statistic in test set $\mathcal{T}(X)$ is $\sum_{z \in \mathcal{X}_h} s(X, z)$.
- Nearest Neighbor Similarity - A descriptive statistic that is not centroid-based is the *nearest neighbor similarity*: a test sample's similarity to its most similar training sample. Given a sample x and the training samples $z \in \mathcal{X}$, the nearest neighbor similarity is defined

$$s_{nn}(x) = \max_{z \in \mathcal{X}} s(x, z). \tag{15}$$

The SDA classifier based on nearest neighbor similarity, denoted by *nnSDA*, may be viewed as a generalization of the similarity-based nearest neighbor classifier (1-NN) defined in 1. That classifier labels x with the same class label as its nearest neighbor without making use of any information about its similarity to such nearest neighbor. The nnSDA classifier, on the other hand, classifies x as the class of its nearest neighbor based on a probabilistic model of $s_{nn}(x)$. The probability model is computed with the mean-constrained maximum entropy approach of Section 2.1, which results in exponential solutions. In this case, the constraint is that the mean of the distribution must be the same as the empirical average of the observed nearest neighbor similarities. Denote by $s_{nn,h}(X)$ the random similarity of a random test sample X to its nearest neighbor in class h. For nnSDA, the constraint is written as

$$E_{P(\mathcal{T}(x)|Y=g)}[s_{nn,h}(X)] = \frac{1}{n_g} \sum_{z \in \mathcal{X}_g} s_{nn,h}(z), \tag{16}$$

and the classification rule becomes to classify as the class $\hat{y}$ that solves

$$\arg \min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f, g) \left(\prod_{h=1}^{G} \gamma_{gh} e^{\lambda_{gh} s_{nn,h}(x)} \right) P(Y = g), \tag{17}$$

where the parameters λ_{gh} and γ_{gh} are computed with the same numerical optimization method used for SDA.

As further discussed in the next section, the SDA framework accommodates any desired set of descriptive statistics $\mathcal{T}(x)$: different similarity functions could be mixed, dissimilarities and similarities can be mixed, and so on.

2.2.2 Generative classifier from arbitrary descriptive statistics

Given an arbitrary set of M descriptive statistics $\mathcal{T}(x)$, the same reasoning of Section 2.1 produces a generative similarity-based classifier. First, the assumption is that $\mathcal{T}(x)$ is sufficient information to classify x leads to the classification rule given in (8). Second, for the mth descriptive statistic $T_m(x) \in \mathcal{T}(x)$, $m = 1, ..., M$, one assumes that its mean with respect to the class conditional distribution of $\mathcal{T}(x)$ is equal to the training sample mean:

$$E_{P(\mathcal{T}(x)|g)}[T_m(X)] = \frac{1}{n_g} \sum_{z \in \mathcal{X}_g} T_m(z). \tag{18}$$

Third, given the M×G constraints specified by (18), one estimates the class-conditional distribution to be the maximum entropy distribution,

$$\begin{aligned} \hat{P}(\mathcal{T}(x)|g) &= \prod_{m=1}^{M} \gamma_{gm} e^{\lambda_{gm} T_m(x)} \\ &= \prod_{m=1}^{M} \hat{P}(T_m(x)|g). \end{aligned} \tag{19}$$

Substituting the maximum entropy solution (19) into (8) yields the SDA classification rule: classify x as the class $\hat{y}$ which solves

$$\arg \min_{f=1,\dots,G} \sum_{g=1}^{G} C(f,g) P(g) \prod_{m=1}^{M} \gamma_{gm} e^{\lambda_{gm} T_m(x)} \ . \tag{20}$$

The parameters $\{\lambda_{gm}, \gamma_{gm}\}$ are calculated as in the centroid-based SDA case described in Section 2.1.

2.2.3 Continuous-valued statistics

The generative classification models presented in this chapter can be extended to the case in which the statistics $\mathcal{T}(x)$ are from a continuous set Ω. This will be the case, for example, when using an overlap similarity (e.g. max{$x[i]$, $z[i]$}) with real-valued features, or when the similarity between X and z is the Euclidean distance. Then, the expectation in (18) is a normalized integral over the continuous set of possible similarity values. Let a and b denote the minimum and maximum possible similarity values (and hence the lower and upper bound on the expectation's integral). Then simplifying (18) yields the relationship

$$\frac{e^{\lambda_{gm} b}(\lambda_{gm} b - 1) - e^{\lambda_{gm} a}(\lambda_{gm} a - 1)}{\lambda_{gm}(e^{\lambda_{gm} b} - e^{\lambda_{gm} a})} = \bar{t}_{gm}, \tag{21}$$

where $\bar{t}_{gm} = \frac{1}{n_g}\sum_{z\in\mathcal{X}_g} T_m(z)$. The solution to (21) can be computed numerically. For the special case $a = 0$ and $b = \infty$, the solution is $\lambda_{gm} = -1/\bar{t}_{gm}$.

3. Local SDA

This chapter introduces *local SDA* (Cazzanti & Gupta, 2007), a similarity-based classifier that is both generative and local. An advantage of generative classifiers is their interpretability: classes are modeled by conditional probability distributions which are assumed to have generated the observed data. An advantage of local classifiers it that they reduce the estimation bias problem which affects generative classifiers. Local SDA combines the qualities of both generative and local classifiers.

For the SDA classifier, the class-conditional generative distributions are exponentials that model the similarities between samples - or more generally the descriptive statistics of the sample. The exponentials are the maximum entropy distributions subject to constraints on the mean values of the similarities. However, when the underlying distributions are complex, a particular set of empirical statistics may fail to capture the necessary information about a sample's class membership. In fact, in SDA, constraining the means of the class-conditional distributions may result in too much model bias, just as the QDA model of one Gaussian per class causes model bias (Hastie et al., 2001). In standard metric learning, one way to address the bias problem while retaining the advantages of a generative approach is to form more flexible Gaussian mixture models. In similarity-based learning, mixture models may also be formed; this approach is discussed in Section 4.

Here, the bias in SDA is addressed by using local classifiers in similarity space. In metric learning, one way to avoid the bias problem is to use local classifiers, e.g. k-NN, which classify test samples based on the class labels of their nearest neighbors. Local classifiers do not estimate probabilistic models for the sample classes and consequently lack the interpretability of generative models. Even so, they provide an intuitive framework for classification through the concepts of nearest-neighbor and neighborhood. In this chapter, SDA is applied to a local neighborhood about the test sample. The resulting *local SDA* classifier trades-off model bias and estimation variance depending on the neighborhood size, while retaining the power of a generative classifier. To the author's knowledge, local SDA is the first example of a classifier that is both generative and local. The only arguable contender is the local nearest- mean classifier (Mitani & Hamamoto, 2000, 2006) for metric learning; however that classifier was not proposed as a generative model.

Local SDA is a straightforward variation of SDA. The local SDA classifier model is that all of the relevant information about classifying a test sample x depends only on the k nearest (most similar) training samples to x. Thus, the local SDA classifier computes the descriptive statistics from a neighborhood of a test sample. More specifically, local SDA is a log-linear generative classifier that models the probability distribution of the similarity $s(x, \mu_h)$ between the test sample x and the class centroids $\{\mu_h\}$, just like SDA. Unlike SDA, the class centroids, the class-conditional similarity probability models, and the estimates of the class priors are computed from a neighborhood of the test sample rather than from the entire training set. Thus, the class centroid definition (5) used for SDA still holds for local SDA; one simply redefines $\mathcal{X}_h$ as the subset of the k nearest neighbors from class h. The class priors are estimated using normalized class membership counts of the neighbors of x, that is $\hat{P}$ $(Y = h)$ = $|\mathcal{X}_h|/k$. The mean similarity constraints (9) for the SDA maximum entropy optimization are formally the same for local SDA, except that the mean is computed from the neighbors of test sample x rather than the whole training set. Thus, the optimized parameters λ_{gh} and γ_{gh} are local. Given the set of local class centroids $\{\mu_h\}$, the local class priors $\hat{P}(Y = g)$, and the local class-conditional model parameters γ_{gh} the local SDA classification rule is identical to the SDA rule (12):

$$\arg\max_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g) \left(\prod_{h=1}^{G} \gamma_{gh} e^{\lambda_{gh} s(x,\mu_h)} \right) \hat{P}(Y = g).$$

A problem can occur if the *h*th class has few training samples in the neighborhood of test sample x. In this case, the local SDA model for class h is difficult to estimate. To avoid this problem, if the number of local training samples in any of the classes is very small, for example $n_h < 3$, the local SDA classifier reverts to the local nearest centroid classifier. If $n_h = 0$ so that $\mathcal{X}_h$ is the empty set, then the probability of class h is locally zero, and that class is not considered in the classification rule (12). This strategy enables local SDA to gracefully handle small k and very small class priors.

Local classification algorithms have traditionally been weighted voting methods, including classifying with local linear regression, which can be formulated as a weighted voting method (Hastie et al., 2001). These methods are by their nature non-parametric and their use arises in situations when the available training samples are too few to accurately build class models. On the other hand, it is known that the number of training samples required by nonparametric classifiers to achieve low error rates grows exponentially with the number of features (Mitani & Hamamoto, 2006). Thus, when only small training sets are available, nonparametric classifiers are negatively impacted by outliers. In 2000, Mitani and Hamamoto (Mitani & Hamamoto, 2000, 2006) were the first ones to propose a classifier that is both model-based and local. However, they did not develop it as a local generative method; instead, they proposed the classifier as a local weighted-distance method. Their nearest-means classifier can be interpreted as a local QDA classifier with identity covariances. In experiments with simulated and real data sets, the local nearest-means classifier was competitive with, and often better than, nearest neighbor, the Parzen classifier, and an artificial neural network, especially for small training sets and for high dimensional problems.

Local nearest-means differs from local SDA in several aspects. First, the classifier by Mitani and Hamamoto in (Mitani & Hamamoto, 2006) learns a metric problem, not a similarity problem: the class prototypes are the local class-conditional means of the features and a weighted Euclidean distance is used to classify a test sample as the class of its nearest class mean. Second, the neighborhood definition is different than the usual k nearest neighbors: they select k nearest neighbors from each class, so that the total neighborhood size is $k \times G$.

More recently, it was proposed to apply a support vector machine to the k nearest neighbors of the test sample (Zhang et al., 2006). The SVM-KNN method was developed to address the robustness and dimensionality concerns that a²ict nearest neighbors and SVMs. Similarly to the nearest-means classifier, the SVM-KNN is a hybrid local and global classifier developed to mitigate the high variance typical of nearest neighbor methods and the curse-of-dimensionality. However, unlike the nearest means classifier of Mitani and Hamamoto, which is rooted in Euclidean space, the SVM-KNN can be used with any similarity function, as it assumes that the class information about the samples is captured by their pairwise similarities without reference to the underlying feature space. Experiments on benchmark datasets using various similarity functions showed that SVM-KNN outperforms k-NN and its variants especially for cases with small training sets and large number of classes. SVM-KNN differs from local SDA because it is not a generative classifier.

Finally, note that different definitions of neighborhood may be used with local SDA. One could use the Mitani and Hamamoto (Mitani & Hamamoto, 2006) definition described above, or radius-based definitions. For example, the neighborhood of a test sample x may be defined as all the samples that fall within a factor of $1+\alpha$ of its similarity to its most similar neighbor, and α is cross-validated. This work employs the traditional definition of neighborhood, as the k nearest neighbors.

3.1 Consistency of the local SDA classifier

Generative classifiers with a finite number of model parameters, such as QDA or SDA, will not asymptotically converge to the Bayes classifier due to the model bias. This section shows that, like k-NN, the local SDA classifier is consistent such that its expected classification error $E[L]$ converges to the Bayes error rate L^* under the usual asymptotic assumptions that the number of training samples $N \rightarrow \infty$, the neighborhood size $k \rightarrow \infty$, but that the neighborhood size grows relatively slowly such that $k{=}N \rightarrow 0$. First a lemma is proven that will be used in the proof of the local SDA consistency theorem. Also, the known result that k-NN is a consistent classifier is reviewed in terms of similarity.

Let the similarity function be $s : \mathcal{B} \times \mathcal{B} \rightarrow \Omega$, where $\Omega \subset \mathbb{R}$ is discrete and let the largest element of -Ω be termed s_{max}. Let X be a test sample and let the training samples $\{X_1,X_2, ... ,X_N\}$ be drawn identically and independently. Re-order the training samples according to decreasing similarity and label them $\{Z_1,Z_2, ..., Z_N\}$ such that Z_k is the kth most similar neighbor of X.

Lemma 1 *Suppose* $s(x,Z) = s_{max}$ *if and only if* $x = Z$ *and* $P(s(x,Z) = s_{max}) > 0$ *where* Z *is a random training sample. Then* $P(s(x,Z_k) = s_{max}) \rightarrow 1$ *as* $k, N \rightarrow \infty$ *and* $k/N \rightarrow 0$.

Proof: The proof is by contradiction and is similar to the proof of Lemma 5.1 in (Devroye et al., 1996). Note that $s(x,Z_k) \neq s_{max}$ if and only if

$$\frac{1}{N}\sum_{i=1}^{N} I_{\{s(x,Z_i)=s_{max}\}} < \frac{k}{N}, \tag{22}$$

because if there are less than k training samples whose similarity to x is s_{max}, the similarity of the kth training sample to x cannot be s_{max}. The left-hand side of (22) converges to $P(s(x,Z) = s_{max})$ as $N\rightarrow\infty$ with probability one by the strong law of large numbers, and by assumption $P(s(x,Z) = s_{max}) > 0$. However, the right-hand side of (22) converges to 0 by assumption. Thus, assuming $s(x,Z_k) \neq s_{max}$ leads to a contradiction in the limit. Therefore, it must be that $s(x,Z_k) = s_{max}$.

Theorem 1 *Assume the conditions of Lemma 1. Define* L *to be the probability of error for test sample* X *given the training sample and label pairs* $\{(Z_1, Y_1), (Z_2, Y_2), ... , (Z_N, Y_N)\}$, *and let* L^* *be the Bayes error*. If $k,N \rightarrow \infty$ *and* $k/N \rightarrow 0$, *then for the local SDA classifier* $E[L] \rightarrow L^*$.

Proof: By Lemma 1, $s(x,Z_i) = s_{max}$ for $i \leq k$ in the limit as $N \rightarrow \infty$, and thus in the limit the centroid μ_h of the subset of the k neighbors that are from class h must satisfy $s(x, \mu_h) = s_{max}$, for every class h which is represented by at least one sample in the k neighbors. By definition of the local SDA algorithm, any class $\overline{h}$ that does not have at least one sample in the k neighbors is assigned the class prior probability $P(Y = \overline{h}) = 0$, so it is effectively eliminated from the possible classification outcomes. Then, the constraint (9) on the expected value of the class-conditional similarity for every class g that is represented in the k neighbors of x is

$$E_{P(s(x,\mu_h)|Y=g)}[s(X, \mu_h)] = s_{max}, \tag{23}$$

which is solved by the pmf $P(s(x, \mu_h) \mid Y = g) = 1$ if $s(x, \mu_h) = s_{max}$, and zero otherwise. Thus the local SDA classifier (12) becomes

$$\hat{y} = \arg \max_{g=1,\ldots G} \hat{P}(Y = g), \tag{24}$$

where the estimated probability of each class $\hat{P}$ $(Y = g)$ is calculated using a maximum likelihood estimate of the class probabilities for the neighborhood. Then, $\hat{P}$ $(Y = g) \rightarrow P(Y = g \mid x)$ as $k \rightarrow\infty$ with probability one by the strong law of large numbers. Thus the local SDA classifier converges to the Bayes classifier, and the local SDA average error $E[L] \rightarrow L^*$.

The known result that k-NN is a consistent classifier can be stated in terms of similarity as a direct consequence of Lemma 1:

Lemma 2 *Assume the conditions of Lemma 1 and define L and L* as in Theorem 1. For the similarity-based k-NN classifier* $E[L] \rightarrow L^*$.

Proof. It follows directly from Lemma 1 that within the size-k neighborhood of x, $Z_i = x$ for $i \leq k$. Thus, the k-NN classifier (1) estimates the most frequent class among the k samples maximally similar to x:

$$\begin{aligned} \hat{y} &= \arg \max_{g=1,\ldots,G} \sum_{i=1}^{k} I(Y_i = g) \\ &= \hat{P}(Y = g). \end{aligned}$$

The summation converges to the class prior $P(Y = g \rightarrow x)$ as $k \rightarrow\infty$ with probability one by the strong law of large numbers, and the k-NN classifier becomes that in (24). Thus the similarity-based k-NN classifier is consistent.

4. Mixture SDA

Like LDA and QDA, basic SDA may be too biased if the similarity space - or more generally the descriptive statistics space - is multi-modal. In analogy to metric space mixture models, the bias problem in similarity space may be alleviated by generalizing the SDA formulation with similarity-based mixture models. In the *mixture SDA* models, the class-conditional probability distribution of the descriptive statistics $\mathcal{T}(x)$ for a test sample x is modeled as a weighted sum of exponential components. Generalizing the single centroid-based SDA classifier and drawing from the metric mixture models (Duda et al., 2001; Hastie et al., 2001), each class h is characterized by c_h centroids $\{\mu_{hl}\}$. The descriptive statistics for test sample x are its similarities to the centroids of class h, $\{s(x, \mu_{h1}), s(x, \mu_{h2}), \ldots, s(x, \mu_{hc_h})\}$, for each class h. The mixture SDA model for the probability of the similarities, assuming that test sample x is drawn from class g, is written as

$$P(s(x,\mu_{h1}), s(x,\mu_{h2}), \ldots, s(x,\mu_{hc_h}) | Y = g) = \sum_{l=1}^{c_h} w_{ghl}\gamma_{ghl}e^{\lambda_{ghl}s(x,\mu_{hl})}, \qquad (25)$$

where $\sum_{l=1}^{c_h} w_{ghl} = 1$ and $w_{ghl} > 0$. Then, the SDA classification rule (12) for mixture SDA becomes to classify x as the class $\hat{y}$ that solves the maximum a posteriori problem

$$\arg\max_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g) \left(\prod_{h=1}^{G} \sum_{l=1}^{c_h} w_{ghl}\gamma_{ghl}e^{\lambda_{ghl}s(x,\mu_{hl})} \right) P(Y = g). \qquad (26)$$

Note how the mixture SDA generative model (25) parallels the metric mixture formulation of Gaussian mixture models (GMMs), with the exponentials $\gamma_{ghl}e^{\lambda_{ghl}s(x,\mu_{hl})}$ in place of the Gaussian components. However, there are deep differences between mixture SDA and metric mixture models. In metric learning, the mixtures model the underlying generative probability distributions of the features. Due to the curse of dimensionality, high-dimensional, multi-modal feature spaces require many training samples for robust model parameter estimation. For example, for d features, GMMs require that a $d \times 1$ mean vector and a $d \times d$ covariance matrix be estimated for each component in each class, for a total of $c_h \times (d^2 + 3d)/2$ parameters per mixture. Constraining each Gaussian covariance to be diagonal, at the cost of an increased number of mixture components, alleviates the robust estimation problem, but does not solve it (Reynolds & Rose, 1995).

When relatively few training samples are available, robust parameter estimation becomes particularly di±cult. In similarity-based learning the modeled quantity is the similarity of a sample to a class centroid. The estimation problem is essentially univariate and reduces to estimating the exponent λ_{ghl} in each component of the mixture, for a total of $c_h \times G \times 2$ parameters per mixture (the scaling parameter γ_{ghl} follows trivially). This simpler classifier architecture allows robust parameter estimation from smaller training set depending on the number of centroids per class, or, more generally, the number of descriptive statistics.

Another major difference between mixture SDA and metric mixture models is in the number of class-conditional probability models that must be estimated. In metric learning, G mixtures are estimated, one for each of the G possible classes from which a sample x may be drawn. In mixture SDA, G^2 mixture models are estimated. Each sample x is hypothesized drawn from class $g = 1, 2, \ldots G$, and its similarities to each of the G classes are modeled by the mixture (25), with $h = 1, 2. \ldots G$. When the number of classes grows, or when the number of components in each mixture model grows, the quadratic growth in the number of needed models presents a challenge in robust parameter estimation, especially when the number of available training samples is relatively small. However, this problem is mitigated by the fact that the component SDA parameters may be robustly estimated with smaller training sets than in metric mixture models due to the simpler, univariate estimation problem at the heart of SDA classification. The next section discusses the mixture SDA parameter estimation procedure.

4.1 Estimating the parameters for mixture SDA models

Computing the SDA mixture model for the similarities of samples $x \in \mathcal{X}_g$ to class h requires estimating the number of components c_h, the component centroids $\{\mu_{hl}\}$, the component weights $\{w_{ghl}\}$ and the component SDA parameters $\{\lambda_{ghl}\}$ and $\{\gamma_{ghl}\}$. This section describes an EM algorithm for estimating these mixture parameters. The algorithm parallels the EM approach for estimating GMM parameters (Duda et al., 2001; Hastie et al., 2001); it is first summarized below, and then explained in detail in the following sections.

Let $\theta_{gh} = \{\{w_{ghl}\}, \{\gamma_{ghl}\}, \{\lambda_{ghl}\}\}$ for $l = 1, 2 \ldots c_h$ be the set of parameters for the class h mixture model to be estimated under the assumption that the training samples z_i, for $i = 1, 2, \ldots n_g$ are drawn identically and independently. Denote by C a random component of the mixture and by $P(C = l \,|\, s(z_i, \mu_{hl}), \theta_{gh})$ the responsibility (Hastie et al., 2001) of the lth component for the ith training sample similarity $s(z_i, \mu_{hl})$. Also write $P(s(z_i, \mu_{hl}) \,|\, C = l, \theta_{gh}) = \gamma_{ghl} e^{\lambda_{ghl} s(z_i, \mu_{hl})}$.

The proposed EM algorithm for mixture SDA is:

1. Compute the centroids $\{\mu_{hl}\}$ with K-medoids algorithm.
2. Initialize the parameters $\{w_{ghl}\}$ and the components $P(s(z_i, \mu_{hl}) \,|\, C = l, \theta_{gh})$.
3. E step: compute the responsibilities

$$P(C = l|s(z_i, \mu_{hl}), \theta_{gh}) = \frac{w_{ghl} P(s(z_i, \mu_{hl})|C = l, \theta_{gh})}{\sum_{l=1}^{c_h} w_{ghl} P(s(z_i, \mu_{hl})|C = l, \theta_{gh})}. \tag{27}$$

4. M step: compute model parameters
   (a) Find the λ_{ghl} which solves

$$E_{P(\mathcal{T}(x)|Y=g)}[s(X, \mu_{hl})] = \frac{\sum_{i=1}^{n_g} s(z_i, \mu_{hl}) P(C = l|s(z_i, \mu_{hl}), \theta_{gh})}{\sum_{i=1}^{n_g} P(C = l|s(z_i, \mu_{hl}), \theta_{gh})}. \tag{28}$$

   (b) Compute the corresponding scaling factor

$$\gamma_{ghl} = \frac{1}{\sum\limits_{s(X,\mu_{hl}) \in \Omega} e^{\lambda_{ghl} s(X, \mu_{hl})}}. \tag{29}$$

   (c) Compute the component weights

$$w_{ghl} = \frac{1}{n_g} \sum_{i=1}^{n_g} P(C = l|s(z_i, \mu_{hl}), \theta_{gh}). \tag{30}$$

5. Repeat E and M steps until convergence criterion is satisfied.

Note that, just like EM for GMMs, the EM algorithm for mixture SDA involves iterating the E step, which estimates the responsibilities, and the M step, which estimates the parameters that maximize the expected log-likelihood of the training data. At each iteration of the M step, the explicit expression (30) updates the component weights. However, unlike EM for GMMs, the update expression for the component parameters (28) is implicit and must be solved numerically. Another difference between the GMM and SDA EM algorithms is in how the centroids are estimated. For GMMs, the component means $\{u_{hl}\}$, which are the metric centroids, are updated at each iteration of the M step. For mixture SDA, the centroids $\{\mu_{hl}\}$ are estimated at the beginning of the algorithm and kept constant throughout the iterations.

The update expressions for the mixture SDA parameters are derived from the expression of the expected log-likelihood of the observed similarities. A standard assumption in EM is that the observed data are independent and identically distributed given the class and mixture component. For mixture SDA, this assumption means that the training sample similarities $\{\mathcal{T}_g(z_i)\} = \{s(z_i, \mu_{hl})\}$, $z_i \in \mathcal{X}_g$ to the component centroids are identically distributed and conditionally independent given the lth class component. Then, the expected log-likelihood of $\{\mathcal{T}_g(z_i)\}$ is

$$L(\{\mathcal{T}_g(z_i)\}|\theta_{gh}) = \sum_{i=1}^{n_g}\sum_{h=1}^{G}\sum_{l=1}^{c_h} \log\left(w_{ghl}\gamma_{ghl}e^{\lambda_{ghl}s(z_i,\mu_{hl})}\right) P(C=l|s(z_i,\mu_{hl}),\theta_{gh}). \quad (31)$$

Using the properties of the logarithm and rearranging the terms, $L(\{\mathcal{T}_g(z_i)\} \mid \theta_{gh})$ splits into the terms depending on w_{ghl} and the terms depending on λ_{ghl} and γ_{ghl}:

$$\begin{aligned} L(\{\mathcal{T}_g(z_i)\}|\theta_{gh}) &= \sum_{i=1}^{n_g}\sum_{h=1}^{G}\sum_{l=1}^{c_h} \log(w_{ghl})P(C=l|s(z_i,\mu_{hl}),\theta_{gh}) \\ &+ \sum_{i=1}^{n_g}\sum_{h=1}^{G}\sum_{l=1}^{c_h} \log(\gamma_{ghl})P(C=l|s(z_i,\mu_{hl}),\theta_{gh}) \\ &+ \sum_{i=1}^{n_g}\sum_{h=1}^{G}\sum_{l=1}^{c_h} \lambda_{ghl}s(z_i,\mu_{hl})P(C=l|s(z_i,\mu_{hl}),\theta_{gh}). \end{aligned} \quad (32)$$

The standard EM approach to maximizing (32) is to set its partial derivatives with respect to the parameters to zero and solve the resulting equations. This is the approach adopted here for estimating the mixture SDA parameters θ_{gh} for all g, h.

The derivation of the expression for the component weights $\{w_{ghl}\}$ follows directly from (32); both the derivation of and the final expression for the component weights are identical to the metric mixtures case. Section 4.1.1 re-derives the well-known expression for w_{ghl}.

Applying the EM approach, however, does not lead to explicit expressions for $\{\lambda_{ghl}\}$ and $\{\gamma_{ghl}\}$. Instead, it leads to many single-parameter constraint expressions for the mean similarities of the training data to the mixture component centroids. These expressions are solved with the same numerical solver used in the single-centroid SDA classifier.

4.1.1 Estimating the component weights

To compute the log-likelihood-maximizing weights w_{ghl}, one uses the standard technique of taking the derivative of the log-likelihood with respect to w_{ghl}, setting it to zero, and solving the resulting expression for w_{ghl}. The constraint $\sum_{l=1}^{c_h} w_{ghl} = 1$ is taken into account with the Lagrange multiplier η:

$$\frac{\partial}{\partial w_{ghl}}\left\{L(\{\mathcal{T}_g(z_i)\}|\theta_{gh}) + \eta\left(\sum_{l=1}^{c_h} w_{ghl} - 1\right)\right\} = \sum_{i=1}^{n_g}\frac{1}{w_{ghl}}P(C=l|s(z_i,\mu_{hl}),\theta_{gh}) + \eta = 0,$$

which gives the well-known expression for the component weights of a mixture model in terms of the responsibilities:

$$w_{ghl} = \frac{1}{n_g} \sum_{i=1}^{n_g} P(C = l | s(z_i, \mu_{hl}), \theta_{gh}). \tag{33}$$

4.1.2 Estimating γ_{ghl} and λ_{ghl}

The same approach used for estimating the component weights $\{w_{ghl}\}$ is adopted to estimate the SDA parameters $\{\gamma_{ghl}\}$ and $\{\lambda_{ghl}\}$: Find the likelihood-maximizing values of the parameters by setting the corresponding partial derivatives to zero and solving the resulting equations. First, since each γ_{ghl} is simply a scaling factor that ensures that each mixture component is a probability mass function, one rewrites

$$\gamma_{ghl} = \frac{1}{\sum_{s(X,\mu_{hl})\in\Omega} e^{\lambda_{ghl} s(X,\mu_{hl})}}, \tag{34}$$

where $X \in \mathcal{X}_g$ is a random sample from class g, $s(X, \mu_{hl})$ is its corresponding random similarity to component centroid μ_{hl}, and Ω is the set of all possible similarity values. Substituting (34) into (32), setting the partial derivative of $L(\{\mathcal{T}_h(z_i)\} \mid \theta_{gh})$ with respect to λ_{ghl} to zero, and rearranging the terms gives

$$\frac{\sum_{s(X,\mu_{hl})\in\Omega} s(X, \mu_{hl}) e^{\lambda_{ghl} s(X,\mu_{hl})}}{\sum_{s(X,\mu_{hl})\in\Omega} e^{\lambda_{ghl} s(X,\mu_{hl})}} \sum_{i=1}^{n_g} P(C = l | s(z_i, \mu_{hl}), \theta_{gh}) = \sum_{i=1}^{n_g} s(z_i, \mu_{hl}) P(C = l | s(z_i, \mu_{hl}), \theta_{gh}). \tag{35}$$

The first term on the left side of (35) is simply the definition of the expected value of the similarity of samples in class g to the lth centroid of class h. Thus, one rewrites (35)

$$E_{P(\mathcal{T}(x)|Y=g)}[s(X, \mu_{hl})] = \frac{\sum_{i=1}^{n_g} s(z_i, \mu_{hl}) P(C = l | s(z_i, \mu_{hl}), \theta_{gh})}{\sum_{i=1}^{n_g} P(C = l | s(z_i, \mu_{hl}), \theta_{gh})}. \tag{36}$$

Expression (36) is an equality constraint on the expected value of the similarity of samples $z_i \in \mathcal{X}_g$ to the component centroids μ_{hl} of class h. This is the same type of constraint that must be solved in the mean-constrained, maximum entropy formulation of single-centroid SDA (9). In (9), the mean similarity of samples from class g to the single centroid of class h is constrained to be equal to the observed average similarity. Analogously, in (36), the mean similarity of the samples from class g to the lth centroid of class h is constrained to be equal to the weighted sum of the observed similarities, where each similarity is weighted by its normalized responsibility. To solve for λ_{ghl}, one uses the same numerical procedure used to solve (9) and described in Section 2.1. Thus, solving for all the $\{\lambda_{ghl}\}$ requires solving the $G \times \sum_{h=1}^{G} c_h$ expressions of (36).

It is not surprising that taking the EM approach to estimating λ_{ghl} has lead to the same expressions for the mean constraints in the maximum entropy approach to density estimation. It is known that maximum likelihood (ML) - the foundation for EM - and maximum entropy are dual approaches to estimating distribution parameters which lead to the same unique solution based on the observed data (Jordan, 20xx). The ML approach assumes exponential distributions for the similarities, maximizes the likelihood, and arrives at constraint expressions whose solutions give the desired values for the parameters. The maximum entropy approach assumes the constraints, maximizes the entropy, and arrives at exponential distributions whose parameters satisfy the given constraints. This powerful dual relationship between ML and maximum entropy extends from metric problems to similarity-based problems; for this reason it leads to the the constraint expression (36), from which λ_{ghl} is numerically computed. The corresponding γ_{ghl} is found by applying (34).

4.1.3 Estimating the centroids

Estimating the centroids of a mixture model encompasses two problems: estimating the number of components (i.e. centroids) $\{c_h\}$, and estimating the centroids $\{\mu_{hl}\}$. This work adopts the common metric learning practice of cross-validating the number of mixture components $\{c_h\}$. The centroids $\{\mu_{hl}\}$ are estimated with the K-medoids algorithm (Hastie et al., 2001), using the maximum-sum-similarity criterion (3). The initial centroids are selected randomly from the training set samples $z_i \in \mathcal{X}_h$.

4.1.4 Initializing EM for SDA

In this work, the component weights $\{w_{ghl}\}$ are uniformly initialized to $w_{ghl} = 1{=}c_h$ and the components are assigned uniform initial probability $P(s(z_i, \mu_{hl}) \,|\, C = l, \theta_{gh}) = 1/c_h$. This initialization reflects the assumption that initially the mixture components equally contribute to a sample's class-conditional probability: it is the least-assumptive initialization. Another strategy would be to initialize the weights by the fraction of training samples assigned to the clusters which result from estimating the centroids with K-medoids. The component probabilities may also be initialized by estimating the SDA parameters $\{\lambda_{ghl}\}$ and $\{\gamma_{ghl}\}$ from the K-medoids clusters. This is analogous to the GMM initialization strategy based on the results of the K-means algorithm. In practice, the simple uniform initialization works well.

5. Experimental results

SDA, local SDA, mixture SDA, and nnSDA are compared to other similarity-based classifiers in a series of experiments: the tested classifiers are the nearest centroid (NC), local nearest centroid (local NC), k-nearest neighbors (k-NN) in similarity space, condensed nearest neighbor (CNN) (Hastie et al., 2001) in similarity space, and the potential support vector machine (PSVM) (Hochreiter & Obermayer, 2006). When the features underlying the similarity are available, the classifiers are also compared to the naive Bayes classifier (Hastie et al., 2001). The counting similarity (the number of features identically shared by two binary vectors) and the VDM (Stanfill & Waltz, 1986; Cost & Salzberg, 1993; Wilson & Martinez, 1997) similarities are used to compute the similarities on which the classifiers operate, except for cases in which similarity is provided as part of benchmark datasets.

The first set of comparisons involves simulated binary data, where each class is generated by random perturbations of one or two centroids. The *perturbed centroids* simulation is a scenario where each class is characterized by one or two prototypical samples (centroids), but samples have random perturbations that make them different from their class centroid in some features. Thus, this simulation fits the centroid- based SDA models, in that each class is defined by perturbations around one or two prototypical centroids.

Then, three benchmark datasets are investigated: the protein dataset, the voting dataset, and the sonar dataset. The results on the simulated and benchmark datasets show that the proposed similarity-based classifiers are effective in classification problems spanning several application domains, including cases when the similarity measures do not possess the metric properties usually assumed for metric classifiers and when the underlying features are unavailable.

For local SDA and local NC, the class prior probabilities are estimated as the empirical frequency of each class in the neighborhood; for SDA, mixture SDA, nnSDA, NC, and CNN they are estimated as the empirical frequency of each class in the entire training data set. The k-NN classifier is implemented in the standard way, with the neighborhood defined by the test sample's k most similar training samples, irrespective of the training samples class. Ties are broken by assigning a test sample to class one.

5.1 Perturbed centroids

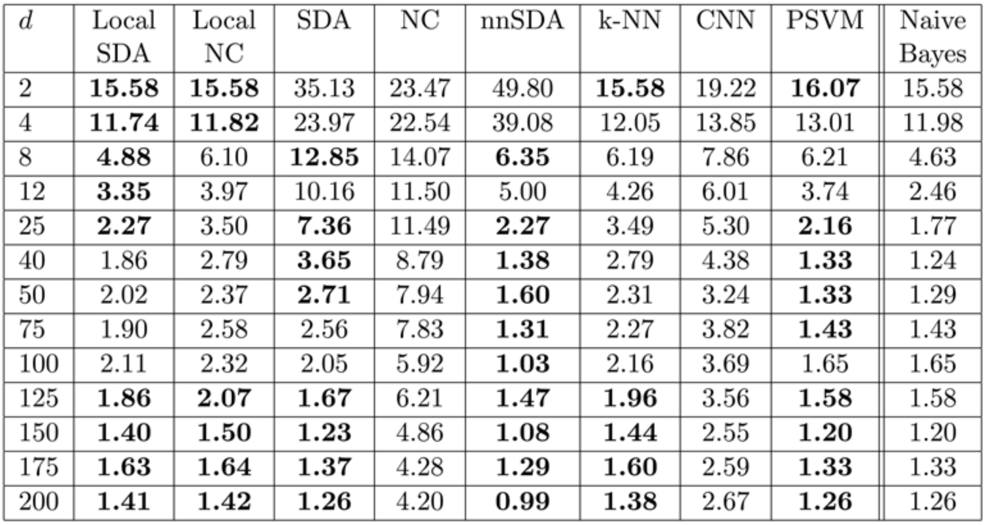
In this two-class simulation, each sample is described by d binary features such that $B = \{0, 1\}^d$. Each class is defined by one or two prototypical sets of features (one or two centroids). Every sample drawn from each class is a class centroid with some features possibly changed, according to a feature perturbation probability. Several variants of the simulation are presented, using different combinations of number of class centroids, feature perturbation probabilities, and similarity measures. Given samples $x, z \in B$, $s(x, z)$ is either the counting or the VDM similarity. The simulations span several values for the feature dimensions d and are run several times to better estimate mean error rates. For each run of the simulation and for each number of features considered, the neighborhood size k for local SDA, local NC, and k-NN is determined independently for the three classifiers by leave-one-out cross-validation on the training set of 100 samples; the range of tested values for k is $\{1, 2, \ldots 20, 29, 39, \ldots , 99\}$. The optimum k is then used to classify 1000 test samples. Similarly, the candidate numbers of components for mixture SDA and for CNN are $\{2, 3, 4, 5, 7, 10\}$. To keep the experiment run time within a manageable practical limit, five-fold cross validation was used to determine the number of components for mixture SDA, and the mixture SDA EM algorithm was limited to 30 iterations for each cross-validated mixture model. The parameters for the PSVM classifier are cross-validated over the range of possible values $\epsilon = \{0.1, 0.2, \ldots 1\}$ and $C = \{1, 51, 101, \ldots 951\}$.

The perturbed centroid simulation results are in Tables 1-8. For each value of d, the lowest mean cross-validation error rate is in bold. Also in bold for each d are the error rates which are not statistically significantly different from the lowest mean error rate, as determined by the Wilcoxon signed rank test for paired differences, with a significance level of 0.05. The naive Bayes classifier results are also included for reference.

5.1.1 Perturbed centroids – one centroid per class

Each class is generated by perturbing one centroidal sample. There are two, equally likely classes, and each class is defined by one prototypical set of d binary features, c_1 or c_2, where c_1 and c_2 are each drawn uniformly and independently from $\{0, 1\}^d$. A training or test sample z drawn from class g has the ith feature $z[i] = c_g[i]$ with probability $1 - p_g$, and $z[i] \neq c_g[i]$ with perturbation probability p_g. In one set of simulation results $p_1 = 1/3$ and $p_2 = 1/30$; thus, class two is well-clustered around its generating centroid and the two classes are well-separated. In another set of simulation results, p_1 = 1⁄3 and p_2 = 1⁄4 and the two classes are not as well separated. Classifiers are trained on 100 training samples and tested on 1000 test samples per run; twenty runs are executed for a total of 20, 000 test samples. The number of features d ranges from d = 2 to d = 200 in the simulation, but the number of training samples is kept constant at 100, so that d = 200 is a sparsely populated feature space. This procedure was repeated for the counting and for the VDM similarities, so there are four sets of results for the one centroid simulation, depending on the perturbation probabilities and the similarity measure used. The results are in Tables 1-4.

The performance of all classifiers increases as d increases. For large d, the feature space is sparsely populated by the training and test samples, which are segregated around their corresponding generating centroids. This leads to good classification performance for all classifiers. For small d, the feature space is densely populated by the samples, and the two classes considerably overlap, negatively affecting the classification performance.

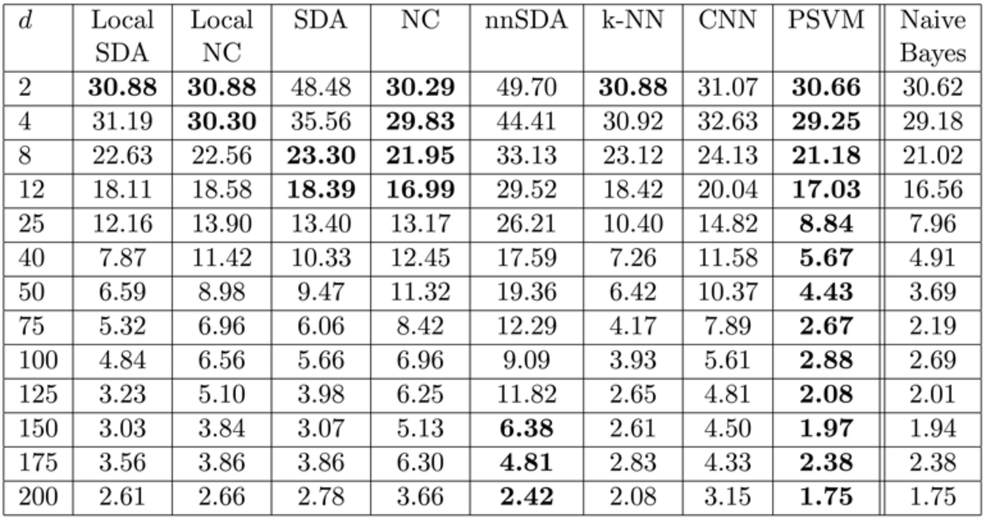


d	Local SDA	Local NC	SDA	NC	nnSDA	k-NN	CNN	PSVM	Naive Bayes
2	**15.58**	**15.58**	35.13	23.47	49.80	**15.58**	19.22	**16.07**	15.58
4	**11.74**	**11.82**	23.97	22.54	39.08	12.05	13.85	13.01	11.98
8	**4.88**	6.10	**12.85**	14.07	**6.35**	6.19	7.86	6.21	4.63
12	**3.35**	3.97	10.16	11.50	5.00	4.26	6.01	3.74	2.46
25	**2.27**	3.50	**7.36**	11.49	**2.27**	3.49	5.30	**2.16**	1.77
40	1.86	2.79	**3.65**	8.79	**1.38**	2.79	4.38	**1.33**	1.24
50	2.02	2.37	**2.71**	7.94	**1.60**	2.31	3.24	**1.33**	1.29
75	1.90	2.58	2.56	7.83	**1.31**	2.27	3.82	**1.43**	1.43
100	2.11	2.32	2.05	5.92	**1.03**	2.16	3.69	1.65	1.65
125	**1.86**	**2.07**	**1.67**	6.21	**1.47**	**1.96**	3.56	**1.58**	1.58
150	**1.40**	**1.50**	**1.23**	4.86	**1.08**	**1.44**	2.55	**1.20**	1.20
175	**1.63**	**1.64**	**1.37**	4.28	**1.29**	**1.60**	2.59	**1.33**	1.33
200	**1.41**	**1.42**	**1.26**	4.20	**0.99**	**1.38**	2.67	**1.26**	1.26

Table 1. Perturbed centroids experiment - One centroid per class. Misclasssification percentage for **counting** similarity, perturbation probabilities $\mathbf{p_1 = 1/3}$ and $\mathbf{p_2 = 1/30}$.

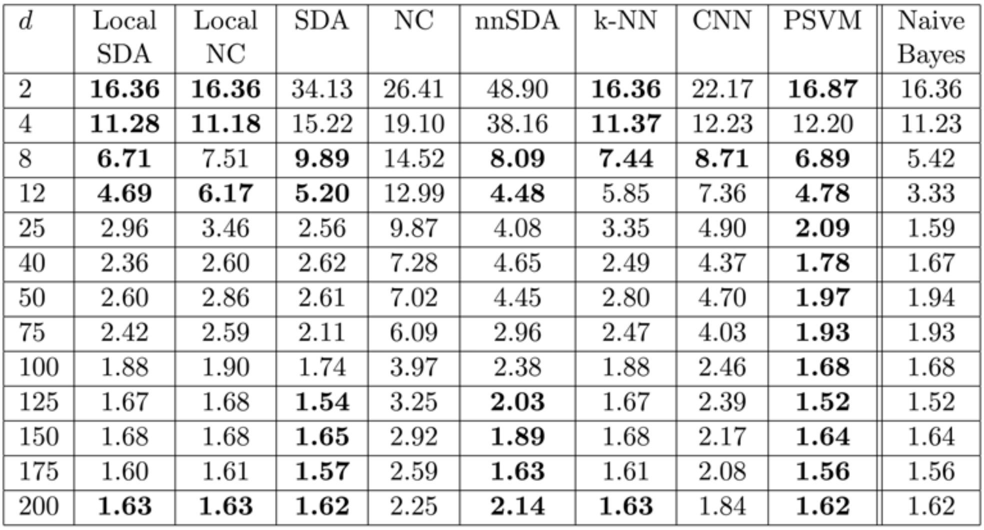
Across all four sets of results, the naive Bayes classifier almost always gives the best performance. Its assumption that the features are independent captures the true underlying relationship of the sample features makes the naive Bayes classifier well suited for these particular data sets: indeed the samples are generated as random vectors of independent binary features. The consequent excellent performance of the naive Bayes classifier provides a reference point for the other classifiers. More generally, when a classification problem involves samples natively embedded in an Euclidean space, as in these perturbed centroids experiments, metric-space classifiers like naive Bayes can perform well. In these cases, the similarity-based classification framework provides no clear advantage.

On the other hand, naive Bayes cannot be used when the samples are not described by vectors of independent features, either because the features are not known, the independence assumption is too restrictive for effective performance, or because the Euclidean representation does not sufficiently capture the pairwise relationships of the samples. In these cases, the similarity-based techniques provide solutions to classification problems. Thus, in these perturbed centroids experiments, the naive Bayes classifier is a good reference for assessing the effectiveness of the similarity-based classifiers, but it is not considered for the Wilcoxon significance tests because it is not generally applicable to similarity-based classification.



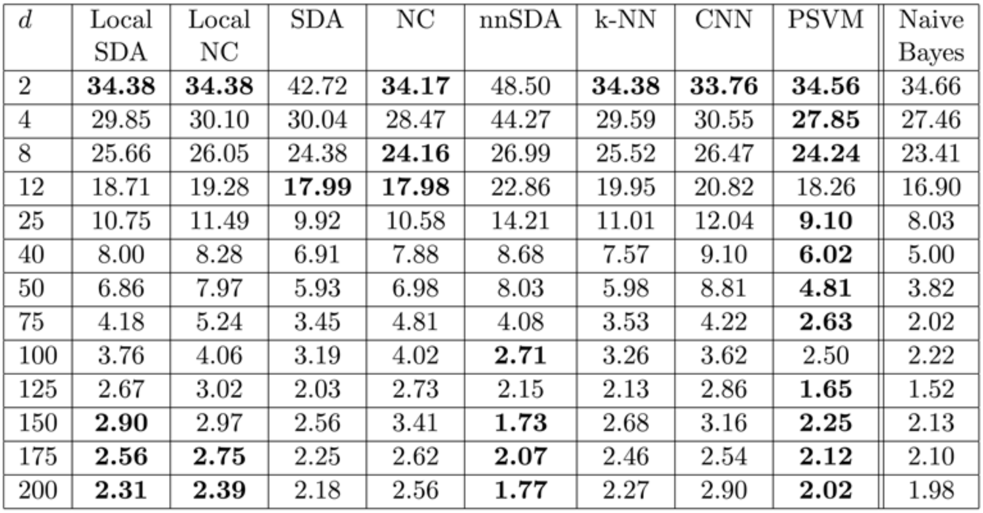
d	Local SDA	Local NC	SDA	NC	nnSDA	k-NN	CNN	PSVM	Naive Bayes
2	**30.88**	**30.88**	48.48	**30.29**	49.70	**30.88**	31.07	**30.66**	30.62
4	31.19	**30.30**	35.56	**29.83**	44.41	30.92	32.63	**29.25**	29.18
8	22.63	22.56	**23.30**	**21.95**	33.13	23.12	24.13	**21.18**	21.02
12	18.11	18.58	**18.39**	**16.99**	29.52	18.42	20.04	**17.03**	16.56
25	12.16	13.90	13.40	13.17	26.21	10.40	14.82	**8.84**	7.96
40	7.87	11.42	10.33	12.45	17.59	7.26	11.58	**5.67**	4.91
50	6.59	8.98	9.47	11.32	19.36	6.42	10.37	**4.43**	3.69
75	5.32	6.96	6.06	8.42	12.29	4.17	7.89	**2.67**	2.19
100	4.84	6.56	5.66	6.96	9.09	3.93	5.61	**2.88**	2.69
125	3.23	5.10	3.98	6.25	11.82	2.65	4.81	**2.08**	2.01
150	3.03	3.84	3.07	5.13	**6.38**	2.61	4.50	**1.97**	1.94
175	3.56	3.86	3.86	6.30	**4.81**	2.83	4.33	**2.38**	2.38
200	2.61	2.66	2.78	3.66	**2.42**	2.08	3.15	**1.75**	1.75

Table 2. Perturbed centroids experiment - One centroid per class. Misclasssification percentage for **counting** similarity, perturbation probabilities $\mathbf{p_1} = \mathbf{1/3}$ and $\mathbf{p_2} = \mathbf{1/4}$.



d	Local SDA	Local NC	SDA	NC	nnSDA	k-NN	CNN	PSVM	Naive Bayes
2	**16.36**	**16.36**	34.13	26.41	48.90	**16.36**	22.17	**16.87**	16.36
4	**11.28**	**11.18**	15.22	19.10	38.16	**11.37**	12.23	12.20	11.23
8	**6.71**	7.51	**9.89**	14.52	**8.09**	**7.44**	**8.71**	**6.89**	5.42
12	**4.69**	**6.17**	**5.20**	12.99	**4.48**	5.85	7.36	**4.78**	3.33
25	2.96	3.46	2.56	9.87	4.08	3.35	4.90	**2.09**	1.59
40	2.36	2.60	2.62	7.28	4.65	2.49	4.37	**1.78**	1.67
50	2.60	2.86	2.61	7.02	4.45	2.80	4.70	**1.97**	1.94
75	2.42	2.59	2.11	6.09	2.96	2.47	4.03	**1.93**	1.93
100	1.88	1.90	1.74	3.97	2.38	1.88	2.46	**1.68**	1.68
125	1.67	1.68	**1.54**	3.25	**2.03**	1.67	2.39	**1.52**	1.52
150	1.68	1.68	**1.65**	2.92	**1.89**	1.68	2.17	**1.64**	1.64
175	1.60	1.61	**1.57**	2.59	**1.63**	1.61	2.08	**1.56**	1.56
200	**1.63**	**1.63**	**1.62**	2.25	**2.14**	**1.63**	1.84	**1.62**	1.62

Table 3. Perturbed centroids experiment - One centroid per class. Misclasssification percentage for **VDM** similarity, perturbation probabilities $\mathbf{p_1} = \mathbf{1/3}$ and $\mathbf{p_2} = \mathbf{1/30}$.



d	Local SDA	Local NC	SDA	NC	nnSDA	k-NN	CNN	PSVM	Naive Bayes
2	**34.38**	**34.38**	42.72	**34.17**	48.50	**34.38**	**33.76**	**34.56**	34.66
4	29.85	30.10	30.04	28.47	44.27	29.59	30.55	**27.85**	27.46
8	25.66	26.05	24.38	**24.16**	26.99	25.52	26.47	**24.24**	23.41
12	18.71	19.28	**17.99**	**17.98**	22.86	19.95	20.82	18.26	16.90
25	10.75	11.49	9.92	10.58	14.21	11.01	12.04	**9.10**	8.03
40	8.00	8.28	6.91	7.88	8.68	7.57	9.10	**6.02**	5.00
50	6.86	7.97	5.93	6.98	8.03	5.98	8.81	**4.81**	3.82
75	4.18	5.24	3.45	4.81	4.08	3.53	4.22	**2.63**	2.02
100	3.76	4.06	3.19	4.02	**2.71**	3.26	3.62	2.50	2.22
125	2.67	3.02	2.03	2.73	2.15	2.13	2.86	**1.65**	1.52
150	**2.90**	2.97	2.56	3.41	**1.73**	2.68	3.16	**2.25**	2.13
175	**2.56**	**2.75**	2.25	2.62	**2.07**	2.46	2.54	**2.12**	2.10
200	**2.31**	**2.39**	2.18	2.56	**1.77**	2.27	2.90	**2.02**	1.98

Table 4. Perturbed centroids experiment - One centroid per class. Misclasssification percentage for **VDM** similarity, perturbation probabilities $\mathbf{p_1} = \mathbf{1/3}$ and $\mathbf{p_2} = \mathbf{1/4}$.

With few exceptions the PSVM performs best on the four sets of results on a wide range of *d*. This is likely because the PSVM classifies a test sample based on its similarities to the entire training set. In contrast, local methods such as local SDA, local NC, nnSDA, k-NN, and CNN make use of a subset of the training samples and thus have less information available to classify. Global methods based on the similarity-to-class-centroid summary statistic such as SDA, NC, and CNN also use less information. It is plausible that the ability to make use of all the similarity information in the training set and to optimally weight the similarities to the training samples gives the PSVM a performance advantage over the other techniques. However, in spite of this advantage, the results show that for low and high values of *d* the SDA-based techniques yield statistically equivalent performance to the PSVM, and in some cases match or exceed its results. When the PSVM statistically produces significantly different results from the other techniques, its performance does not hugely surpass them. Thus the similarity-based techniques possess the ability to produce good classification results using less information. This quality can be immensely useful when few training samples are available.

In all four sets of results, the SDA-based algorithms generally perform better than their non-generative counterparts: local SDA performs better than local NC and SDA performs better than NC. This shows that generative models based on the similarity of samples to local or global class centroids provide increased discriminative power over the non-generative centroid-based similarity models. Furthermore, in almost all cases across the four sets of results, local SDA performs better than SDA. While the classification performance of SDA is good, its inherent model bias prevents it from achieving even better performance; local SDA is not as susceptible to model bias, and is able to perform very well. Still, the SDA performance is close to that of the local SDA in all cases and sometimes it surpasses it (VDM similarity with $p_2 = 1/4$), a confirmation that the single-centroid generative model at the heart of SDA matches well the perturbed single-centroid experimental setup for these sets of results.

The similarity-space k-NN performs well, albeit not as well as the PSVM. Compared to SDA, k-NN performs better only for the counting similarity and $p_2 = 1/4$. Since SDA matches well the class models for the generated samples, it is not surprising that it performs better than k-NN, which does not rely on class models. However, k-NN does better when the class two perturbed samples are more likely to differ from their generating class two centroid ($p_2 = 1/4$), that is when the classes overlap more. In this case, it is more di±cult to estimate the class centroids, and the SDA performance is affected. On the other hand, SDA is better than k-NN for the VDM similarity, for both $p_2 = 1/30$ and $p_2 = 1/4$. The VDM similarity is calculated from class-dependent lookup tables pre-computed from the training set, and this additional information seems to favor the SDA classifier more than the k-NN. Local SDA, performs slightly better than k-NN when $p_2 = 1/30$ for both counting and VDM similarities.

The CNN classifier generally does not perform as well as k-NN. This is expected, because, as for its metric learning analog, the condensing process primarily aims to reduce the size of large training sets and possibly eliminate outliers rather than to improve classification performance. The observed lower performance of CNN compared to k-NN reflects the expectation that classification performance will degrade when using the condensed training set instead of the full set of available training samples.

The nnSDA classifier performs well for the counting similarity when $p_2 = 1/30$, and in general for higher values of d. For low values of d the performance is particularly poor: for $d = 2$ the error rate is essentially equal to that of a random classifier (50%) and for $d = 4$ it is only slightly better. In fact, the nnSDA performance is limited by the interplay of its asymptotic behavior and the value of d. Recall that by Lemma (1) from Section 3.1, $P(s(x,Z_k) = s_{max}) \to 1$ as $k,N \to\infty$ and $k/N \to 0$, where k is the neighborhood size, N is the number of available training samples, and Z_k is the k-th nearest neighbor of test sample x. Then, it follows that $P(s_{nn,h}(x) = s_{max}) \to 1$ for all h as $k, n \to\infty$, because $s_{nn,h}(x) = s(x,Z_1)$ for $Z_1 \in \mathcal{X}_h$ as $k\to\infty$. Thus, for nnSDA, the similarities of a test sample to its nearest neighbors in each class are all identical in the limit of infinite number of training samples. Consequently, for a large training set, all class discriminants in the nnSDA classification rule (17) are identical and therefore uninformative. The classification rule (17) reduces to the trivial rule that classifies according to the cost-adjusted class priors,

$$\hat{y} = \arg \min_{f=1,\ldots,G} \sum_{g=1}^{G} C(f,g)P(Y=g). \tag{37}$$

When 0-1 costs are used, as in this simulation, the rule (37) always classifies as the class g with the highest prior probability $\hat{P}$ ($Y = g$), estimated as the empirical frequency from the training data:

$$\hat{y} = \arg \max_{g=1,\ldots,G} \hat{P}(y=g). \tag{38}$$

In this experiment, the samples are generated from two, a priori equally likely classes, so the limit misclassification rate is $1 - \max_{g} \hat{P}(Y=g) \approx 0.5.$

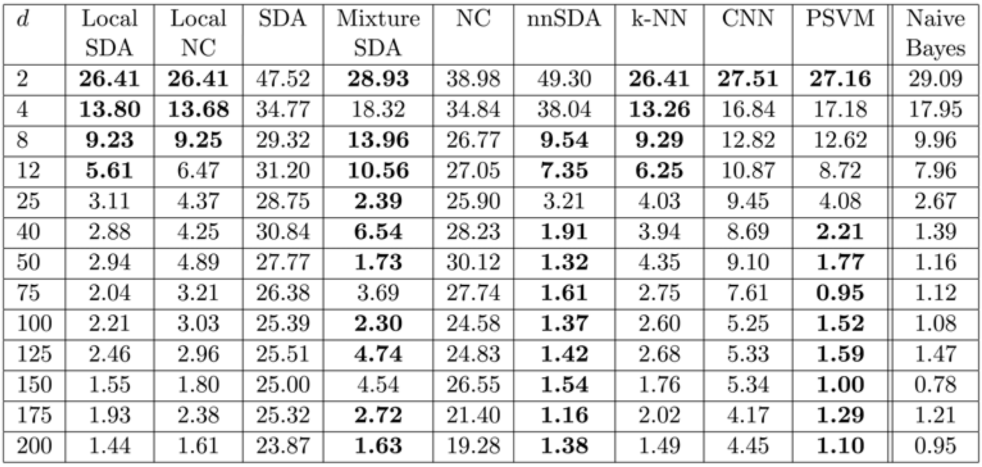
The limit error rate is noticeable when d is small. In this case the similarity can take on values in a limited range bounded by d ($s(x, z) \in [0, 1 \ldots d]$ for the counting similarity) and the training set is highly redundant. Thus, a test sample x is very likely to be maximally similar to its nearest neighbor from each class, and $s_{nn,h}(x)$ is uninformative. In higher dimensions, the experimental results show that the training set is sufficiently sparse for effective classification. Thus nnSDA is a viable classifier for sparse training sets which do not cover the entire range of possible values for the chosen similarity. In applications when few training samples are available, nnSDA can be a valuable tool for achieving actionable classification results.

5.1.2 Perturbed centroids – two centroids per class

In this variation of the perturbed centroids simulation, each class is characterized by two prototypical samples, c_{11}, c_{12} for class one, and c_{21}, c_{22} for class two. Each time the simulation is run, the centroids c_{11}, c_{12}, c_{21}, c_{22} are drawn independently and identically using a uniform distribution over $\mathcal{B}$.

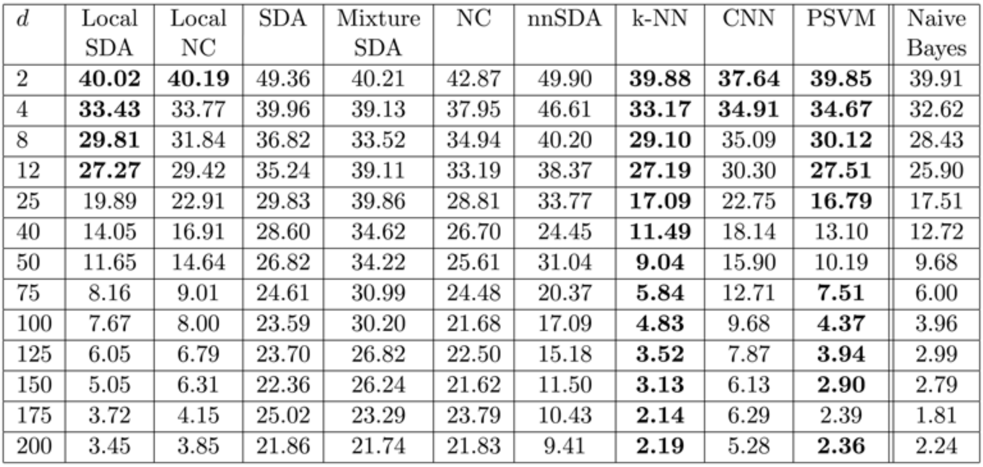
Every sample drawn from each class is a perturbed version of one of the two class prototypes, where the class labels are drawn independently and identically with probability 1/2. A training or test sample z drawn from class one is randomly selected to be $z = c_{11}$ or $z = c_{12}$ with probability 1/2, and then for each $i = 1, \ldots, d$, z's ith feature is probabilistically perturbed so that $z[i] \neq c_{11}[i]$ with probability p_{11} (or $z[i] \neq c_{12}[i]$ with probability p_{12}). Thus on average, a randomly drawn sample based on c_{11} will have dp_{11} features that are different from the class prototype c_{11}'s features. Likewise, a training or test sample v drawn from class two starts out as $v = c_{21}$ or $v = c_{22}$ with probability 1/2, but then for each $i = 1, \ldots, d$, v's ith feature is changed so that $v[i] \neq c_{21}[i]$ with probability p_{21} (or $v[i] \neq c_{22}[i]$ with probability p_{22}).

The number of features d ranges from $d = 2$ to $d = 200$ in the simulation, but the number of training samples is kept constant at 100, so that $d = 200$ is a sparsely populated feature space. Two different sets of values of the perturbation probabilities p_{11}, p_{12}, p_{21}, p_{22} were used: in the first case $p_{11} = p_{12} = 1/3$ and $p_{21} = p_{22} = 1/30$, so that the class two samples are much more tightly clustered around c_{21} and c_{22} than the class one samples are with respect to c_{11} and c_{12}. In the second case, $p_{11} = p_{12} = 1/3$ and $p_{21} = p_{22} = 1/4$, resulting in a higher Bayes error. Each simulation was run twenty times, for a total of 20,000 test samples. The resulting mean error rates are given in Tables 5-8.



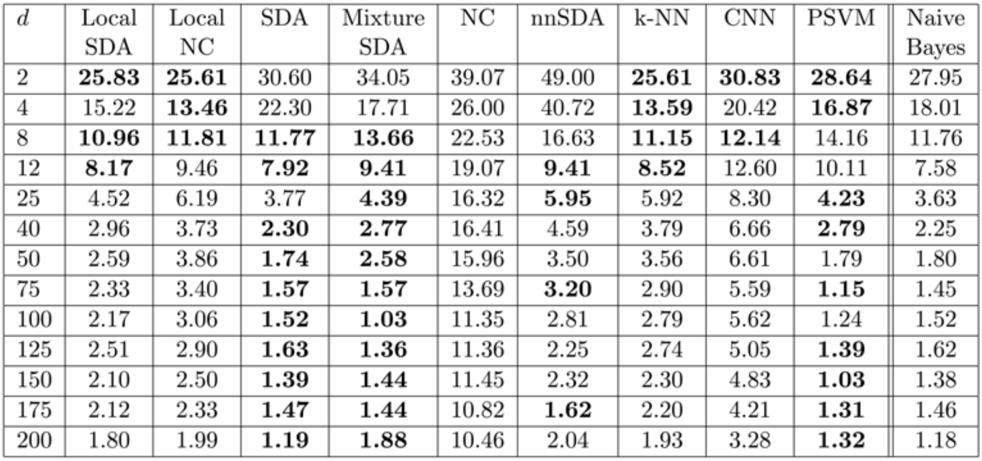
d	Local SDA	Local NC	SDA	Mixture SDA	NC	nnSDA	k-NN	CNN	PSVM	Naive Bayes
2	**26.41**	**26.41**	47.52	**28.93**	38.98	49.30	**26.41**	**27.51**	**27.16**	29.09
4	**13.80**	**13.68**	34.77	18.32	34.84	38.04	**13.26**	16.84	17.18	17.95
8	**9.23**	**9.25**	29.32	**13.96**	26.77	**9.54**	**9.29**	12.82	12.62	9.96
12	**5.61**	6.47	31.20	**10.56**	27.05	**7.35**	**6.25**	10.87	8.72	7.96
25	3.11	4.37	28.75	**2.39**	25.90	3.21	4.03	9.45	4.08	2.67
40	2.88	4.25	30.84	**6.54**	28.23	**1.91**	3.94	8.69	**2.21**	1.39
50	2.94	4.89	27.77	**1.73**	30.12	**1.32**	4.35	9.10	**1.77**	1.16
75	2.04	3.21	26.38	3.69	27.74	**1.61**	2.75	7.61	**0.95**	1.12
100	2.21	3.03	25.39	**2.30**	24.58	**1.37**	2.60	5.25	**1.52**	1.08
125	2.46	2.96	25.51	**4.74**	24.83	**1.42**	2.68	5.33	**1.59**	1.47
150	1.55	1.80	25.00	4.54	26.55	**1.54**	1.76	5.34	**1.00**	0.78
175	1.93	2.38	25.32	**2.72**	21.40	**1.16**	2.02	4.17	**1.29**	1.21
200	1.44	1.61	23.87	**1.63**	19.28	**1.38**	1.49	4.45	**1.10**	0.95

Table 5. Perturbed centroids experiment - Two centroids per class. Misclassification percentage for **counting** similarity, perturbation probabilities $\mathbf{p}_{11} = \mathbf{p}_{12} = \mathbf{1/3}$ and $\mathbf{p}_{21} = \mathbf{p}_{22} = \mathbf{1/30}$.



d	Local SDA	Local NC	SDA	Mixture SDA	NC	nnSDA	k-NN	CNN	PSVM	Naive Bayes
2	**40.02**	**40.19**	49.36	40.21	42.87	49.90	**39.88**	**37.64**	**39.85**	39.91
4	**33.43**	33.77	39.96	39.13	37.95	46.61	**33.17**	**34.91**	**34.67**	32.62
8	**29.81**	31.84	36.82	33.52	34.94	40.20	**29.10**	35.09	**30.12**	28.43
12	**27.27**	29.42	35.24	39.11	33.19	38.37	**27.19**	30.30	**27.51**	25.90
25	19.89	22.91	29.83	39.86	28.81	33.77	**17.09**	22.75	**16.79**	17.51
40	14.05	16.91	28.60	34.62	26.70	24.45	**11.49**	18.14	13.10	12.72
50	11.65	14.64	26.82	34.22	25.61	31.04	**9.04**	15.90	10.19	9.68
75	8.16	9.01	24.61	30.99	24.48	20.37	**5.84**	12.71	**7.51**	6.00
100	7.67	8.00	23.59	30.20	21.68	17.09	**4.83**	9.68	**4.37**	3.96
125	6.05	6.79	23.70	26.82	22.50	15.18	**3.52**	7.87	**3.94**	2.99
150	5.05	6.31	22.36	26.24	21.62	11.50	**3.13**	6.13	**2.90**	2.79
175	3.72	4.15	25.02	23.29	23.79	10.43	**2.14**	6.29	2.39	1.81
200	3.45	3.85	21.86	21.74	21.83	9.41	**2.19**	5.28	**2.36**	2.24

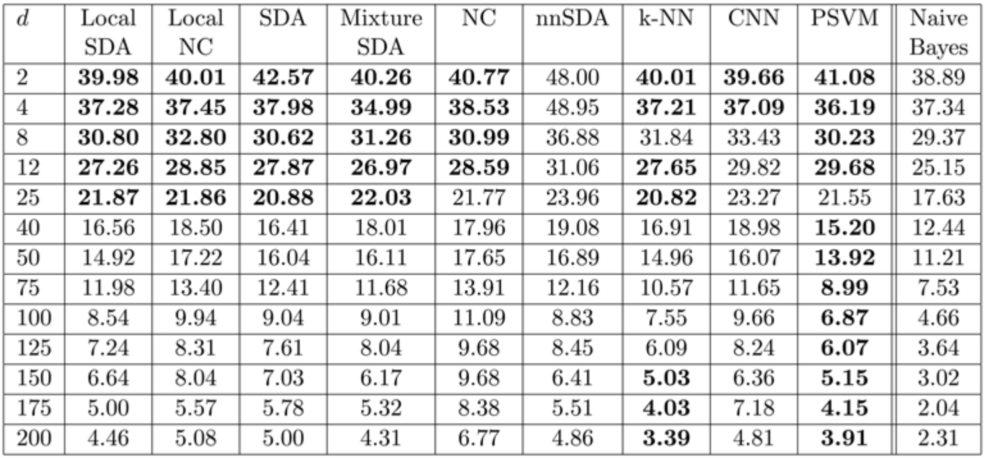
Table 6. Perturbed centroids experiment - Two centroids per class. Misclassification percentage for **counting** similarity, perturbation probabilities $\mathbf{p}_{11} = \mathbf{p}_{12} = \mathbf{1/3}$ and $\mathbf{p}_{21} = \mathbf{p}_{22} = \mathbf{1/4}$.



d	Local SDA	Local NC	SDA	Mixture SDA	NC	nnSDA	k-NN	CNN	PSVM	Naive Bayes
2	**25.83**	**25.61**	30.60	34.05	39.07	49.00	**25.61**	**30.83**	**28.64**	27.95
4	15.22	**13.46**	22.30	17.71	26.00	40.72	**13.59**	20.42	**16.87**	18.01
8	**10.96**	**11.81**	**11.77**	**13.66**	22.53	16.63	**11.15**	**12.14**	14.16	11.76
12	**8.17**	9.46	**7.92**	**9.41**	19.07	**9.41**	**8.52**	12.60	10.11	7.58
25	4.52	6.19	3.77	**4.39**	16.32	**5.95**	5.92	8.30	**4.23**	3.63
40	2.96	3.73	**2.30**	**2.77**	16.41	4.59	3.79	6.66	**2.79**	2.25
50	2.59	3.86	**1.74**	**2.58**	15.96	3.50	3.56	6.61	1.79	1.80
75	2.33	3.40	**1.57**	**1.57**	13.69	**3.20**	2.90	5.59	**1.15**	1.45
100	2.17	3.06	**1.52**	**1.03**	11.35	2.81	2.79	5.62	1.24	1.52
125	2.51	2.90	**1.63**	**1.36**	11.36	2.25	2.74	5.05	**1.39**	1.62
150	2.10	2.50	**1.39**	**1.44**	11.45	2.32	2.30	4.83	**1.03**	1.38
175	2.12	2.33	**1.47**	**1.44**	10.82	**1.62**	2.20	4.21	**1.31**	1.46
200	1.80	1.99	**1.19**	**1.88**	10.46	2.04	1.93	3.28	**1.32**	1.18

Table 7. Perturbed centroids experiment - Two centroids per class. Misclassification percentage for **VDM** similarity, perturbation probabilities $\mathbf{p}_{11} = \mathbf{p}_{12} = \mathbf{1/3}$ and $\mathbf{p}_{21} = \mathbf{p}_{22} = \mathbf{1/30}$.

For all four sets of results, the local SDA classifier performs better than the local NC classifier. This result agrees with the analogous case for the single centroid experiments and attests to the advantage that similarity-based generative models provide over simpler nearest-centroid classifiers. However, the SDA classifier yields better classification than its counterpart NC classifier only for the VDM similarity. For the counting similarity, SDA does not provide an advantage over NC. There are two causes that contribute to this outcome. First, the single-centroid SDA is a biased model that does not match the true two-centroids-per-class experimental setup. Consider class one and its centroids, c_{11} and c_{12}. SDA at best correctly estimates one of the two centroids per class, let's say $\hat{c}_{11}$. Thus, the estimated centroid- based generative model for class one is a good match for the samples which are generated as random perturbations of c_{11}. The model, however, is not a good match for samples generated as random perturbations of c_{12}. The model cannot distinguish the similarities of these class one samples to $\hat{c}_{11}$ from their similarities to the centroids of class two. The result is that the c_{12}-generated samples are classified according to the class priors, that is half as class one and half as class two. The same argument applies to class two, so that overall about 25% of the samples are misclassified. Indeed, the SDA error rates quickly settle to ≈25% for the counting similarity for medium to large values of d. For lower d, the class overlap due to the density of the feature space dominates the misclassification rate.



d	Local SDA	Local NC	SDA	Mixture SDA	NC	nnSDA	k-NN	CNN	PSVM	Naive Bayes
2	**39.98**	**40.01**	**42.57**	**40.26**	**40.77**	48.00	**40.01**	**39.66**	**41.08**	38.89
4	**37.28**	**37.45**	**37.98**	**34.99**	**38.53**	48.95	**37.21**	**37.09**	**36.19**	37.34
8	**30.80**	**32.80**	**30.62**	**31.26**	**30.99**	36.88	31.84	33.43	**30.23**	29.37
12	**27.26**	**28.85**	**27.87**	**26.97**	**28.59**	31.06	**27.65**	29.82	**29.68**	25.15
25	**21.87**	**21.86**	**20.88**	**22.03**	21.77	23.96	**20.82**	23.27	21.55	17.63
40	16.56	18.50	16.41	18.01	17.96	19.08	16.91	18.98	**15.20**	12.44
50	14.92	17.22	16.04	16.11	17.65	16.89	14.96	16.07	**13.92**	11.21
75	11.98	13.40	12.41	11.68	13.91	12.16	10.57	11.65	**8.99**	7.53
100	8.54	9.94	9.04	9.01	11.09	8.83	7.55	9.66	**6.87**	4.66
125	7.24	8.31	7.61	8.04	9.68	8.45	6.09	8.24	**6.07**	3.64
150	6.64	8.04	7.03	6.17	9.68	6.41	**5.03**	6.36	**5.15**	3.02
175	5.00	5.57	5.78	5.32	8.38	5.51	**4.03**	7.18	**4.15**	2.04
200	4.46	5.08	5.00	4.31	6.77	4.86	**3.39**	4.81	**3.91**	2.31

Table 8. Perturbed centroids experiment - Two centroids per class. Misclassification percentage for **VDM** similarity, perturbation probabilities $\mathbf{p}_{11} = \mathbf{p}_{12} = \mathbf{1/3}$ and $\mathbf{p}_{21} = \mathbf{p}_{22} = \mathbf{1/4}$.

The second cause contributing to the observed SDA results stems from the way the class centroids are generated. Each class centroid is generated randomly from a multivariate uniform distribution over the feature space. Thus, there is no guarantee that two centroids from the same class be more similar to each other than two centroids from different classes, that is there is no guarantee that $s(c_{1i}, c_{1j}) < s(c_{1i}, c_{2j})$ for $i, j = 1, 2$. On the contrary, on average over many draws from the sample space, the centroids are equally similar, and consequently the samples generated as perturbations of c_{12}, c_{21}, and c_{22} are approximately equally similar to c_{11}. This amplifies the detrimental effect of the bias in the SDA model. If the condition on the similarities between centroids $s(c_{1i}, c_{1j}) < s(c_{1i}, c_{2j})$ were enforced, then even the biased SDA model would produce better classification results.

The performance of mixture SDA is comparable to that of SDA if not slightly better. For the particularly simple case of the counting similarity with $p_{21} = p_{22} = 1/30$, the mixture SDA provides an order of magnitude improvement over SDA, showing that it is able to alleviate the bias problem inherent to the single-centroid SDA. However, in all other perturbed centroids results the comparison between the performance of mixture SDA and SDA is inconclusive. For $p_{21} = p_{22} = 1/4$, the overlap between the classes overshadows any performance gains mixture SDA might obtain; for the VDM results, the advantage provided by the optimized similarity measure brings the performance of SDA and mixture SDA closer together, and thus limits the gains of mixture SDA. Given the increase in complexity of the mixture SDA classifier and its inconclusive performance advantages, for these experiments it might be more advantageous to use local classifiers such as local SDA to obtain improved performance. The results show that local SDA consistently performs very well, and with only a few exceptions outperforms SDA and mixture SDA.

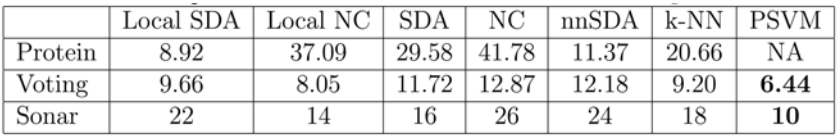
Note that for the VDM similarity, SDA produces excellent classification results which are very competitive with local SDA and local NC, and consistently outperform NC. The large improvement is attributable to the fact that the VDM undergoes a training phase, performed on the training set, in which the class information is used to optimize the similarity measure for class discrimination. This training step greatly benefits the SDA classifier and yields improved classification results for all classifiers when compared to the counting similarity, which does not rely on such pre-computations.

As for the single-centroid results, nnSDA is most effective at higher values of *d*, when the feature space is sparsely populated by the samples. A consistently good performer is the k-NN classifier, which is very competitive with local SDA, local NC, and the PSVM when $p_{21} = p_{22} = 1/30$, and often outperforms them when $p_{21} = p_{22} = 1/4$. Using a subset of the training samples, as with CNN, negatively impacts the classification performance for all sets of simulations, consistently with the single-centroids results discussed in the previous section.

5.2 Benchmark data sets

Three benchmark data sets are used to analyze further the performance of various similarity-based classifiers: a data set of protein similarities, a data set of congressional voting records, and a data set of aural sonar similarities. The tested classifiers are the local SDA, local NC, SDA, NC, nnSDA, *k*-NN, and PSVM classifiers. The mixture SDA and CNN classifiers are not tested on these data sets, as the long time required to cross-validate their parameters does not justify their attainable performance.

The performance of the classifiers on all three benchmark data sets is evaluated as the *leave-one-out error*, as follows. One sample is set aside as the test sample, and all other $N - 1$ samples are used for training. The parameters for each classifier are cross-validated on the $N - 1$ training samples using leave-one-out cross validation. The resulting best parameters are used to train each classifier on the entire $N - 1$ training samples, and the trained classifier finally classifies the test sample. The process is repeated until all available samples are tested by the trained classifiers. For local SDA, local NC and k-NN, the neighborhood size is cross-validated on the set of possible sizes {1, 2, ... 20, 30 ... 100, 150, 200}. The PSVM parameters are cross-validated over the sets of possible values C = {1, 51, ... 951}, and ϵ = {0.1, 0.2, ... 1}. The class priors are estimated to be the empirical probability of seeing a sample from each class, with Laplace correction (Jaynes, 2003). Table 9 shows the percent leave-one-out error for each classifier evaluated on the three benchmark datasets. The data sets experiments are discussed in more detail in the following sections.



	Local SDA	Local NC	SDA	NC	nnSDA	k-NN	PSVM
Protein	8.92	37.09	29.58	41.78	11.37	20.66	NA
Voting	9.66	8.05	11.72	12.87	12.18	9.20	**6.44**
Sonar	22	14	16	26	24	18	**10**

Table 9. Percentage of leave-one-out misclassifications on the protein data set.

5.2.1 Protein data

Many bioinformatics prediction problems are formulated in terms of pairwise similarities or dissimilarities. An example is the protein data set used by (Hochreiter & Obermayer, 2006). For this data set, pairwise dissimilarity values are calculated using the evolutionary distance, which is the probability that an amino acid sequence transforms into another one (Hofmann & Buhmann, 1997). The sample space $\mathcal{B}$ is not enumerated, so classification must be done based only on the pairwise dissimilarity values. The dataset contains 213 proteins with class labels "HA" (72 samples), "HB" (72 samples), "M" (39 samples) , and "G" (30 samples). The SDA, local SDA, nearest centroid, local nearest centroid, and k-NN classifiers natively support multiclass classification problems, so they can be applied directly to this four-class experiment. The PSVM, however, is a binary classifier and cannot be applied to this multiclass data set.

Guessing that all samples were from the most prevalent class would yield a 66.2% error rate. The simple one-centroid per class model of SDA achieves half that error, and works better than the more flexible local nearest centroid classifier. Local SDA, local nearest centroid and k-NN all have the same free parameter, the neighborhood size *k*. Of these, local SDA is seen to be best suited to this problem.

5.2.2 Voting data set

The UCI voting data set (Newman et al., 1998) records the voting record of 435 members of the US House of Representatives on 16 bills. The binary classification problem is to predict each member's political party affiliation given the voting records. Each of the 16 votes is either a yes, a no, or "neither", so there are 16 features which can each take on 3 possible values. This classification problem can be treated as a similarity-based classification problem by applying a similarity function to the trinary feature space. The adopted similarity in this experiment is the counting similarity.

5.2.3 Aural sonar echoes classification

In the sonar echoes classification experiment, the data consist of 100 pairwise similarities assessed by human listeners. The listeners rated the pairwise similarities of digitized active sonar echoes from two classes { clutter or target { without knowledge of the class labels, and based their evaluation of similarity only on their perceptual judgement of how the echoes sounded similar; thus, the underlying features of similarity are inaccessible. Each listener assigned a discrete similarity value between 1 and 5 to each pair of echoes; each pair was rated by two different listeners, and the two assigned similarity scores were added, so that the range of possible values for the similarity is [2, 10]. The target and clutter classes are equally likely, each one containing 50 echoes. This set of echoes is particularly difficult to classify in that metric-space classifiers produced incorrect results. Further details on this data set are in (Philips et al., 2006).

6. Summary

The chapter introduced a new framework for classification that is both *similarity-based* and *generative*: *similarity discriminant analysis*, or *SDA*. The experimental results show that the classifiers resulting from the proposed SDA framework have practical advantages in terms of performance, interpretability, and ease of use. SDA is *similarity-based* in that it classifies samples based on their pairwise similarities and does not require that the samples be described by numerical feature vectors, the standard sample description method in metric learning. SDA is *generative*, in that it estimates probabilistic models based on descriptive statistics of the classes. Having access to probability estimates is important. A probabilistic framework seamlessly accommodates multi-class classifiers, asymmetric misclassification costs, and class priors. Furthermore, probability estimates are easily fused into into larger systems, and can be used to identify abnormal samples that have low probability of any class. The generative models in the SDA family are solutions to constrained maximum entropy problems where the constraints are placed on the mean values of the similarity-based descriptive statistics. As dictated by the principle of maximum entropy, the resulting generative class models are exponential functions of the similarity statistics.

Di®erent choices for the descriptive statistics lead to different SDA classifiers. This chapter focused on the centroid-based SDA classifiers: each class is described by a prototypical sample, a *centroid*, and the generative models are based on the similarities of the samples to each class centroid. SDA accommodates various definitions of centroid; this chapter focused on the maximum-sum-similarity centroid. The nearest neighbor similarity is also explored as a descriptive statistic, yielding the nnSDA classifier.

As with LDA and QDA, the power of the SDA generative classifier depends on how well its model matches the true class-conditional distributions. A mismatched model will be biased and produce erroneous classifications. The centroid-based SDA classifier is a good match for single-centroid distributions of objects, but is a biased model for multi-centroidal distributions. This chapter proposes *local SDA* and *mixture SDA* as similarity-based generative classifiers with reduced bias that can be used for multimodal distributions. Local SDA is the SDA classifier applied to a local neighborhood of a test sample. A local class centroid can be viewed as a representative prototype for the class in the neighborhood of a test sample and the class-conditional models provide an estimate of the local distribution of the similarities to the local centroid. Local SDA was shown to be a Bayes error-consistent classifier and is the first classifier to be similarity-based, generative, *and* local. Mixture SDA builds on the metric-learning mixture models by modeling each class as a linear combination of several single-centroid SDA models. The parameters for the mixture SDA classifier can be estimated with the EM algorithm.

The family of SDA classifiers is very competitive with, and often outperforms, their corresponding non-generative similarity-based classifier. SDA competes with nearest centroid; local SDA competes with local NC. The SDA classifiers are also competitive with the PSVM, the state-of-the-art support vector machine for similarity-based classification. The PSVM bases its classification on the entire training set of pairwise similarities. This requires enumeration of size $N \times N$ similarity matrices, thus posing computational challenges for large data sets. Furthermore, PSVM is a non-generative, intrinsically binary classifier: it is di±cult to view it in a probabilistic framework where there are more than two possible classes for the data samples. The SDA classifiers remain competitive while relying on more parsimonious representations of the underlying similarity relationships between the samples. Furthermore, the generative quality of the SDA family of classifiers provides intuitive information about the similarity characteristics of the data. The SDA-generated probability estimates are useful for interpreting the results in a probabilistic framework, and allow for class priors and costs to be seamlessly integrated into the classification rules.

7. References

S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(4): 509-522, April 2002.

M. Bicego, V. Murino, M. Pelillo, and A. Torsello. Special issue on similarity-based classification. *Pattern Recognition*, 39, October 2006.

L. Cazzanti and M. R. Gupta. Local similarity discriminant analysis. In *Intl. Conf.on Machine Learning (ICML)*, 2007.

L. Cazzanti and M. R. Gupta. Information-theoretic and set-theoretic similarity. In *Proc. of the IEEE Intl. Symposium on Information Theory*, pages 1836-1840, 2006.

L. Cazzanti, M. R. Gupta, and A. J. Koppal. Generative models for similarity-based classification. *Pattern Recognition*, 41, number = 7, pages = 2289-2297, YEAR = 2008,.

S. Cost and S. Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10(1):57-78, 1993.

T. Cover and J. Thomas. *Elements of Information Theory*. John Wiley and Sons, New York, 1991.

L. Devroye, L. Gyorfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag Inc., New York, 1996.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, 2001.

B. S. Everitt and S. Rabe-Hesketh. *The Analysis of Proximity Data*. Arnold, London, 1997.

I. Gati and A. Tversky. Weighting common and distinctive features in perceptual and conceptual judgments. *Cognitive Psychology*, (16):341-370, 1984.

M. R. Gupta, L. Cazzanti, and A. J. Koppal. Maximum entropy generative models for similarity-based learning. In *Proc. IEEE Intl. Symposium on Information Theory*, 2007.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag, New York, 2001.

S. Hochreiter and K. Obermayer. Support vector machines for dyadic data. *Neural Computation*, 18(6):1472-1510, 2006.

T. Hofmann and J.M. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(1), January 1997.

D. W. Jacobs, D. Weinshall, and Y. Gdalyahu. Classification with nonmetric distances: Image retrieval and class representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(6):583-600, June 2000.

E. T. Jaynes. On the rationale for maximum entropy methods. *Proc. of the IEEE*, 70(9):939{952, September 1982.

E. T. Jaynes. *Probability theory: the logic of science*. Cambridge University Press, 2003.

M. I. Jordan. An Introduction to Probabilistic Graphical Models. To be published, 20xx.

W. Lam, C. Keung, and D. Liu. Discovering useful concept prototypes for classification based on filtering and abstraction. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(8):1075-1090, August 2002.

D. Lin. An information-theoretic definition of similarity. *Proc. of the Intl. Conf. on Machine Learning*, 1998.

M. Lozano, J. M. Sotoca, J. S. Sánchez, F. Pla, E. Pekalska, and R. P. W. Duin. Experimental study on prototype optimisation algorithms for prototype-based classification in vector spaces. *Pattern Recognition*, 39:1827-1838, 2006.

MATLAB: The Language of Technical Computing. The MathWorks, Natick, MA, 2006 edition.

Y. Mitani and Y. Hamamoto. Classifier design based on the use of nearest neighbor samples. *Proc. of the Intl. Conf. on Pattern Recognition*, pages 769-772, 2000.

Y. Mitani and Y. Hamamoto. A local mean-based nonparametric classifier. *Pattern Recognition Letters*, 27(10):1151-1159, July 2006.

D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz. UCI repository of machine learning databases, 1998.

E. Pekalska, P. Paclíc, and R. P. W. Duin. A generalized kernel approach to dissimilarity-based classification. *Journal of Machine Learning Research*, pages 175-211, 2001.

E. Pekalska, R. P. W. Duin, and P. Paclík. Prototype selection for dissimilarity-based classifiers. *Pattern Recognition Letters*, 39:189-208, 2006.

S. Philips, J. Pitton, and L. Atlas. Perceptual feature identification for active sonar echoes. In *IEEE OCEANS*, 2006.

D. A. Reynolds and R. C. Rose. Robust text-independent speaker-identification using Gaussian mixture speaker models. *IEEE Trans. on Speech and Audio Processing*, 3(1), 1995.

S. Santini and R. Jain. Similarity measures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(9):871-883, September 1999.

S. Sattath and A. Tversky. On the relation between common and distinctive feature models. *Psychological Review*, (94):16-22, 1987.

G. Schwartz and A. Tversky. On the reciprocity of proximity relations. *Journal of Mathematical Psychology*, 22(3):301-307, September 1980.

P. Simard, Y. Le Cun, and J. Denker. Efficient pattern recognition using a new transformation distance. *Advances in Neural Information Processing Systems 5*, pages 50-68, 1993.

C. Stanfill and D. Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29(12):1213-1228, December 1986.

A. Tversky. Features of similarity. *Psychological Review*, (84):327-352, 1977.

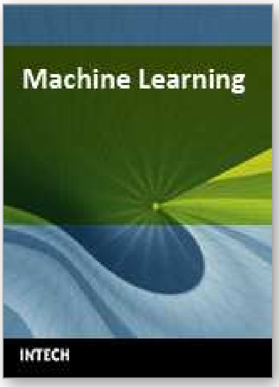
A. Tversky and I. Gati. Studies of similarity. In E. Rosch and B. Lloyd, editors, *Cognition and Categorization*. Earlbaum, Hillsdale, N.J., 1978.

A. Tversky and J. W. Hutchinson. Nearest neighbor analysis of psychological spaces. *Psychological Review*, 93:3-22, 1986.

D. Weinshall, D. W. Jacobs, and Y. Gdalyahu. Classification in non-metric spaces. *Advances in Neural Information Processing Systems 11*, pages 838-844, 1999.

D. R. Wilson and T. R. Martinez. Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research*, 6:1-34, 1997.

H. Zhang, A. C. Berg, M. Maire, and J. Malik. SVM-KNN: discriminative nearest neighbor classification for visual category recognition. *Proc. of the IEEE Conf. o Computer Vision and Pattern Recognition*, pages 2126 - 2136, 2006.


Machine Learning
INTECH