Pattern Recognition Letters 32 (2011) 1523–1531

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

A Bayes-true data generator for evaluation of supervised and unsupervised learning methods

Janick V. Frasch [a,b], Aleksander Lodwich [a], Faisal Shafait [a,*], Thomas M. Breuel [b]

[a] German Research Center for Artificial Intelligence (DFKI), 67663 Kaiserslautern, Germany
[b] University of Kaiserslautern, 67663 Kaiserslautern, Germany

ARTICLE INFO

Article history:
Received 9 October 2009
Available online 6 May 2011
Communicated by S. Sarkar

Keywords:
Synthetic data generation
Benchmarking
Experimental proofs

ABSTRACT

Benchmarking pattern recognition, machine learning and data mining methods commonly relies on real-world data sets. However, there are some disadvantages in using real-world data. On one hand collecting real-world data can become difficult or impossible for various reasons, on the other hand real-world variables are hard to control, even in the problem domain; in the feature domain, where most statistical learning methods operate, exercising control is even more difficult and hence rarely attempted. This is at odds with the scientific experimentation guidelines mandating the use of as directly controllable and as directly observable variables as possible. Because of this, synthetic data possesses certain advantages over real-world data sets. In this paper we propose a method that produces synthetic data with guaranteed global and class-specific statistical properties. This method is based on overlapping class densities placed on the corners of a regular k-simplex. This generator can be used for algorithm testing and fair performance evaluation of statistical learning methods. Because of the strong properties of this generator researchers can reproduce each others experiments by knowing the parameters used, instead of transmitting large data sets.

1. Introduction

From an engineering point of view, the results of evaluations performed on machine learning methods must be transferable to a desired application scenario. The evaluations need to yield a dependency of the performance indicators from basic problem characteristics in order to be able to generalize the results from the initially restricted test environment.

Such problem characteristics in the domain of machine learning predominantly concern the data. This has been theoretically indicated by the No-Free-Lunch theorem discussed by Wolpert and Macready (1997) and practically verified many times, most prominently by the Statlog report in (King et al., 1995). A particular focus on the transferability resulting from the data characteristics has been payed by van der Walt and Bernard (2007), yielding a metaintelligence successfully predicting the appropriate classifiers based on dataset characterizations.

In contrast to this, current practice of experimental algorithm evaluation lacks the extensive use of such data characteristics. Instead, it is dominated by the use of domain specific and mostly real-world datasets (most notably the UCI repository UCI, 2009). Unfortunately, as Rachkovskij and Kussul (1998) point out, there are two major drawbacks in using such data:

- Firstly, the aforementioned basic characteristics of the data (e.g. covariance, Bayes error, geometric aspects, external and intrinsic dimensions) are typically not fully known and extremely difficult to control.
- Secondly, accessing real-world data in most domains is difficult for budget, technical or ethic reasons to name some. This typically results in a limited amount of data available for testing purposes, which can impede and even preclude important practical conclusions.

In order to allow for a more systematic investigation of machine learning methods, yielding generic results, we believe it is necessary to use synthetic data generators that control significant data characteristics.

Given this importance of synthetic data generation, publications on generation methods with controlled statistical properties are quite rare. Indeed, Pei and Zaïane (2006) observed the same:

"*Surprisingly, little work has been done on systematically generating artificial datasets for the analysis and evaluation of data analysis algorithms in data mining area.*"

Motivated by this, we contribute a new synthetic data generator with controlled statistical data characteristics, like means, covari-

[*] Corresponding author.
E-mail addresses: janick.frasch@iwr.uni-heidelberg.de (J.V. Frasch), aleksander.lodwich@dfki.de (A. Lodwich), faisal.shafait@dfki.de (F. Shafait), tmb@cs.uni-kl.de (T.M. Breuel).

0167-8655/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2011.04.010

ance, intrinsic dimensionality and, most importantly, the Bayes error rate.

2. State-of-the-art

In the beginning, a brief survey on existing synthetic data generators shall be given, demonstrating the scope of current research and its limitations. We start with domain-specific approaches, discussing more general approaches afterwards.

2.1. Specific approaches

Most of the synthetic dataset generators are specifically designed to test machine learning methods in specific domains. Such generators use parameterized models that create realistically looking data.

Helmers and Bunke (2003), for example, have built handwriting samples from actual character templates for the testing of handwriting recognition systems using synthetic generators.

Baird (2000) gave a good overview on methods generating synthetic document image data from real images through the use of degradation models. These approaches have parameters controlling the degradation operators, not the statistical properties of the degraded characters.

Other examples of domain specific synthetic data generators are the ones from document image defect models discussed by Baird in (Baird, 1993) and the ones which have been investigated by the US Census Bureau for maintaining the confidentiality of original micro data, cf. Abowd and Lane (2003). The latter generators were also trained from real-world data.

Purely synthetic data, generated from statistical models based on Gaussian mixtures trained on real-world data has been used for evaluation purposes in protein spot detection methods for 2D images, as described by Rogers et al. (2003).

For knowledge discovery systems Jeske et al. (2005) proposed a system for generating synthetic social data based on a to-be-defined semantic graph.

Davidov et al. (2004) described a method for an automated dataset acquisition of labeled textual content from the World Wide Web to be used in text categorization systems. Collecting data according to specific parameters can be seen as a special approach for synthetic data generation.

For completeness, some preceding work to be mentioned is the IBM Quest Data Generator (Srikant, 1999) and the GSTD (Theodoridis et al., 1999). IBM's generator is relational and is targeting at concept learning from relational databases which is a similar work to Jeske et al. (2005); the GSTD is motivated by geodesian applications and simulates Brownian movement of rectangularly grouped vertices through spatio-temporal feature space. None of the methods controls global statistical properties, though.

There is a series of Hidden Markov Model (HMM) based data generators. Unlike the methods presented above hardly any control over the generation can be exercised. This is due to the fact, that although HMM is a quite general process model, final HMM models are trained to be specific, e.g. like a natural source of data. This restricts the value of this method for benchmarking purposes as it follows an objective totally different from the one emphasized in this paper.

2.2. More general approaches

All of the above methods have in common that they only work for specific problem domains where problem dependent variables are known but the resulting feature distributions are quite complex and statistically not fully understood.

Rachkovskij and Kussul (1998) demonstrated a more general algorithm generating samples from a partitioned feature space including a uniform background noise. The proposed method offers parameters for partitioning the feature space into different regions ("classes") as well as for the distribution of the samples generated from each class. One major drawback of the so called *DataGen* method is that it only produces class distributions with many uncontrolled properties, which make it difficult to interpret the generated data as feature vectors from any actual real-world scenario. As the authors acknowledge, generating data with internal dependencies is not directly possible with the presented method, but anyway desirable.

Another example for an attempt to create generic data generators is the work of Pei and Zaïane (2006). The main goal of this work was to generate data usable for unsupervised learning and outlier detection. The parameters are the number of clusters, distance between the clusters, the total number of points, cluster distributions and outlier controlling parameters. The generator operates in 2D and provides five levels of difficulty. The generator can add uniform background noise and transform the densities into a higher dimensional space. However the generator is not able to control statistical properties relevant for the qualitative or quantitative evaluation of supervised learning algorithms.

The work probably most closely related to the one presented here is the one from van der Walt and Bernard (2007). The authors have used very simple Gaussian mixture models in order to debunk myths or "common wisdom" like for example *"discriminative classifiers tend to be more accurate than model-based classifiers at classification tasks"* or that *"k-nearest-neighbors (kNN) classifiers are almost always close to optimal in accuracy, for an appropriate choice of k"*. This work clearly demonstrates the usefulness of synthetic dataset generators based on generic density mixtures. However, the authors had to set the centers and other parameters of the mixtures manually without an explicit model that guarantees a unique mapping between the generated model instance and the characteristic parameters (i.e. parameters that correspond to measurable quantities of the experiments to be executed on the data generated by the model). Such an approach lacks control over the constructed model instance and therefore its reproducibility, since insignificant (from the point of view of later experiments) parameters accounting for additional degrees of freedom need to be set.

Also, due to this overhead of manual work, the authors were only able to consider up to 10 dimensions, 4 classes and 1000 samples/density which are not representative of real classification problems today. In order to continue this promising path of research, a greater degree of automation is desirable.

In the following, we propose a new method for synthetic dataset generation, the White Gaussian on *k*-simplex model (WGKS). The benefit of our contribution is the full control over the relevant statistical properties, which can overcome the hitherto discussed disadvantages. In particular, the Bayes error rate is controlled, which is important as a qualitative reference for the evaluation of machine learning algorithms.

3. The White Gaussians on *k*-simplex data generation model

The purpose of the WGKS generator is to have a simple and symmetric model representing very simple classification problems. Independent of the learning model, these problems are characterized by the equal stochastic treatment of the classes. No class density shall have any advantage of being modeled better than any other class density. This generator is creating unimodal problems for each class where resulting data is stratified in its nature.

In particular the following features of the model will be known and directly controllable: (1) the number of classes, (2) the number of intrinsic dimensions, (3) the number of samples and, most importantly (4) the Bayes error between the class densities. By post-processing, it is also possible to control (5) the centroid, (6) the external dimensionality and (7) the dataset covariance.

The most significant advantage of this method over existing methods is the known Bayes error rate; thus we call this class of methods Bayes-true data generators.

The WGKS generator can be obtained from http://madm.dfki.de/downloads for the *Python* environment using the *NumPy* and *SciPy* libraries.

The generation model in the standard form is based on white Gaussian densities located at the corners of a regular k-simplex (see Fig. 1). It can be modified choosing other distributions instead.

The k-simplex explicitly defines the number of densities and implicitly defines the number of intrinsic dimensions $d = k - 1$. This setup guarantees that all density centers are having the same euclidean distance to each other, thus ensuring the aforementioned equal stochastic treatment.

The central difficulty in generating data according to this model lies in the fact that even in the comparatively easy case of white Gaussian distributions there is no known explicit solution for the edge length λ of the simplex with Bayes error $e(\lambda)$.

This means, the cost function $\|e(\lambda) - e_t\|$, where e_t is the target error rate, has to be minimized over all λ by some optimization procedure.

Furthermore, even $e(\lambda)$ cannot be computed explicitly for arbitrary dimensions. This means standard nonlinear optimization methods will imply repeated Bayes error computation by time-consuming Monte Carlo simulation. To allow a convenient practical usage of WGKS, we will present some major ideas for speedup in the following.

3.1. Computing the Bayes error

The probabilities of each class c for a sample x_h are given by

$$f_c(x_h, \lambda) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(x_h - \lambda v_c)^T \Sigma^{-1} (x_h - \lambda v_c)}, \qquad (1)$$

where Σ is the covariance matrix (can be omitted, because we are considering white Gaussians), λ is the edge length of the simplex and v_c are the coordinates of the vertices of the k-simplex with edge length *1*.

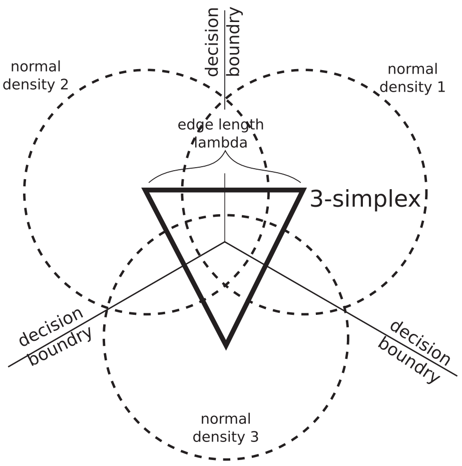

Fig. 1. WGKS places white normal densities on the corners of a k-simplex.

We consider two different Monte Carlo methods for computing the Bayes error:

A) *Direct evaluating of the confusion matrix.* All samples x_h whose originator class c_o is not equal to the most likely class $\hat{c}$ with largest posterior probability $f_c(x_h, \lambda)$ are counted as false classified. The number of false classified samples divided by the overall number of samples is the estimate for the Bayes error rate.

B) *Importance sampling.* The probabilities of each class c, in particular of the most likely class $\hat{c}$ for a sample x_h are given by

$$p(\hat{c}|x_h) = \frac{f_{max}(x_h)}{\sum_{i=0}^{k} f_i(x_h)},$$

with $f_{max}(x_h)$ being the largest density. Computing their mean over all samples $\bar{p} = \frac{1}{|X|}\sum_{h=1}^{|X|} p(\hat{c}|x_h)$ also yields an estimator for the Bayes error rate $e = 1 - \bar{p}$, based on the idea of importance sampling. For a detailed mathematical derivation see Appendix A.

It can be shown (see Appendix A) that the variance of method B is always smaller than the variance of method A, hence on the average method B converges faster.

However, there is a trade-off, since the computation of the likelihoods for each sample in method B usually takes more time as the check for misclassification in method A can be sped up for many kinds of identical densities (e.g. for white Gaussians, this reduces to distance calculations between the sample and the center of each density). Also, method A is inevitable when the posteriors of the samples are not available but only classifier labels of the Bayesian classifier.

3.2. Revision of Bayes error computation

Since the Bayes error computation relies on Monte Carlo simulation, a large number of samples is needed for reasonable accuracy. However, the amount of samples is highly limited by the available computation time and memory. We propose a more efficient approach that exploits the total symmetry of this setup. From the implementation it can be seen that the time complexity of the model with likelihood recomputation based on arbitrary Σ for every investigated λ is $O(l \cdot n \cdot k^3)$ if Σ is the identity matrix (or a scalar multiple thereof) because then Σ can be ignored in (1). The optimized method has a time complexity of $O(l \cdot n \cdot k)$, where l is the number of optimization epochs, n is number of samples per density and k is the number of densities. For a detailed discussion on time complexity we refer to Lodwich et al. (2009).

The basic idea is that the full data space is divided in equal compartments by the decision hyperplanes. The hyperplanes consist of all equidistant points between pairs of centers. Since all density functions are symmetrically arranged and identical, the total Bayes error must be equal to the Bayes error in each compartment. Therefore, we can reduce the problem to computing the Bayes error in a single compartment. As it can be seen from Fig. 2 (for formal details see Theorem 1 in Appendix B.1) the total mass of a density function outside the decision manifolds containing its center equals the mass of the other density functions constituting the error in that compartment. This means that the Bayes error e can be computed as the ratio between the density mass lying outside the boundaries and the total density mass of a single density. Using Monte Carlo estimation this can be approximated by simply counting the ratio between samples lying outside the compartment and the overall number of samples, where the samples are generated from only one density.

In order to avoid computing the likelihoods $f_c(x_h, \lambda)$ for each sample x_h, the density can be transformed in a way that the com-

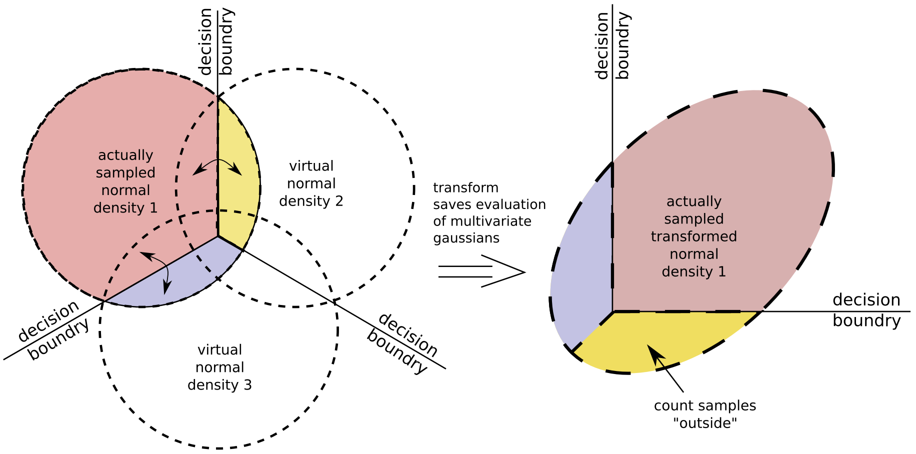

Fig. 2. Exploiting the symmetry of the setup for error computation.

partment containing the center of the density coincides with the first orthant in the Cartesian coordinate system. Thus the task of performing a Bayes optimal classification on a sample x_h is reduced to the task of checking for a negative sign in one of the coordinates of x_h and subsequent manipulations of λ during optimization are reduced to simple vector translation operations. Alternatively, the densities can be resampled and transformed in every optimization step. However, this has two disadvantages: (a) the computational complexity is higher and (b) the cost function is not deterministically monotone.

For a detailed mathematical description of the transformation, see Appendix B.2.

3.3. Accuracy improvement for small error rates

Least-squares retrofitting of experimental data strongly suggests that for a fixed k the relation between λ and e is based on the inverse of the complemented error function (see Fig. 3).

This seems reasonable, since white Gaussian densities are used. By contrast, inverse polynomial and trigonometric functions and combination thereof failed to approximate the recorded values which are presented in Fig. 3.

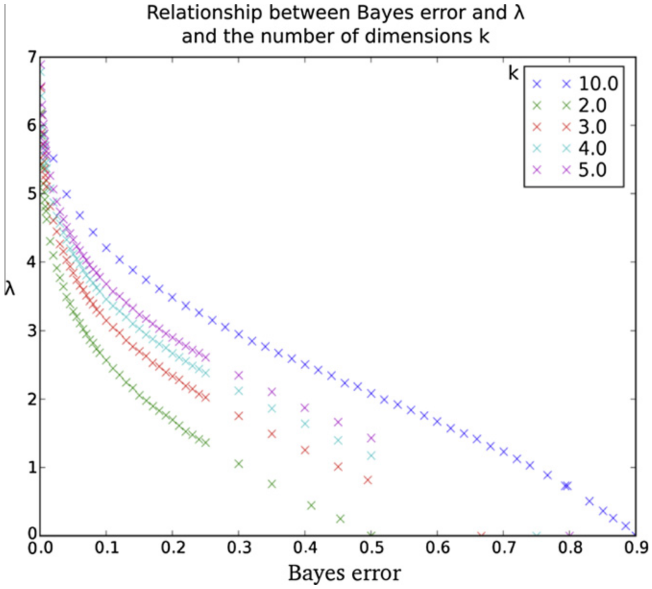

Fig. 3. Relationship between the error and the distance λ for a given number of white normal densities positioned on a regular k-simplex. The implicit number of dimensions is $k-1$. Values are obtained experimentally.

In particular, $\lambda(e,k) = -\frac{1}{s_k}\mathrm{erfcinv}(2e) + y_k$, which only depends on a slope parameter s_k and a vertical shift parameter y_k, presents a well suitable approximation of the λ-e relationship; other parameters proved to be insignificant for the experimental data.

Since all probability density functions are identical, it is obvious that for $\lambda = 0$ the Bayes error rate equals $(k-1)/k$, which yields one parameter.

For determining the second parameter, the calculation of λ for any other target error rate e_t will suffice. We choose $e_t = 0.5$, since the absolute slope of the inverse error function is smallest at this value. This means, a small deviation of the achieved error rate due to the statistical spread of the Monte Carlo estimation will result in the smaller deviation of λ than for any other target error rate.

These considerations can be used in two ways. Firstly, since the variance of λ is larger for small e_t, it can make sense to compute the parameters of the inverse error function instead as suggested and evaluate the latter for the desired e_t.

Secondly, when λ is needed for multiple error rates for the same k, the presented approach only needs to perform the expensive Monte Carlo simulation once.

We used this method to generate a lookup table that is implemented in the downloadable version of the WGKS and can be used for fast determination of λ. The parameter values were computed in the above manner using extensive Monte Carlo sampling. Note that for $k = 2$ the inverse error function is just the standard inverse error function. For $k > 100$, due to computational limitations, a full list of values cannot be provided. However, for these k, λ depends almost log-linear on k (cf. Fig. 4) and hence can be extrapolated.

3.4. Optimization of λ for generic densities

The generator can be modified in order to generate samples from more general density prototypes (again placed on the corners of a regular k-simplex). However, using a look up table method as mentioned above does not work anymore, since the distance to error relationship will typically become non error-function like and may even be not monotonic or discontinuous. Finding the optimal parameter λ in this case can either be performed by gradient descent methods (if the class densities are differentiable), in our case by multisection search (Csallner et al., 2000) or by other global (possibly stochastic) optimization methods.

3.5. Post processing guaranteeing intrinsic dimensionality

In the WGKS module, there is a series of post processing steps that can be used to create affine variations of a dataset in order to test a classifier for its robustness. We propose random transla-

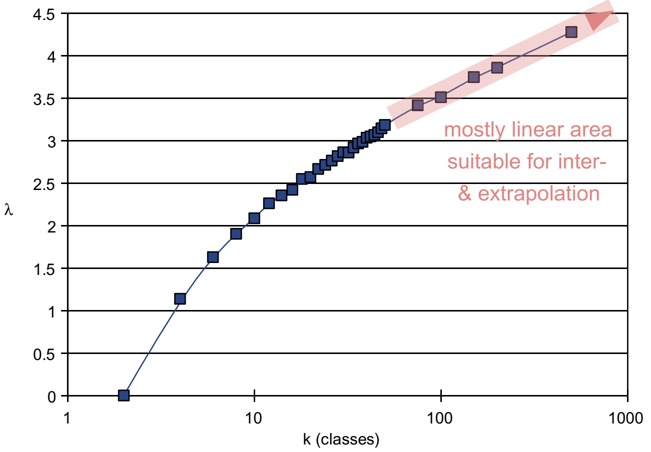

Fig. 4. How λ and k depend on each other (drawn in a logarithmic scaling of the k-axis for 50% error rate).

tion, scaling, shearing and rotation methods, as well as a method of fitting the data to a given covariance matrix. The main purpose of these modifications is to embed the generated data into higher dimensional spaces.

3.5.1. Random translation

Translations do not change any other parameters than the means and are simply implemented by adding a random vector to all samples. For WGKS, this random vector is obtained in two steps:

1. A random direction vector is generated form an d-dimensional normal distribution and subsequent normalization.
2. A random length scalar is generated from a uniform or normal distribution.

3.5.2. Random rotation

Rotation changes the means and the absolute covariance characteristics but the shape is preserved.

In order to obtain a random rotation matrix with each rotation equally likely, we make use of the fact, that each orthogonal matrix with determinant 1 represents a rotation.

We start off with a $d \times d$ matrix with entries $x_{i,j} \sim \mathcal{N}(0,1)$. Since this resembles d d-dimensional normally distributed random vectors, we get an equally distributed likelihood concerning the directions (angles). We then orthogonalize this matrix, e.g. by a QR decomposition, taking Q (which already has determinant 1) and yield an orthogonal matrix with each rotation direction equally likely, since the directions of the starting vectors were uniformly random. This procedure has been described by Genz (1999).

3.5.3. Random scaling

As scaling parameter we chose a log-normal distributed variable $z \sim \mathcal{N}_{log}(0,\sigma)$. According to the survey done by Limpert et al. (2001) this is the most dominant form of one-sided distributions for natural problems. The characteristic of the chosen distribution parameters is that the distribution median is at *1*. The σ-parameter, which is accounting for the spread, is user-defined.

3.5.4. Random shearing

An easy way to obtain a random shearing matrix is to start with a orthogonal matrix I (identity matrix) and break its perpendicularity by modifying l randomly chosen specific linear dependencies between different coordinates by

$$M_{i_\xi,j_\xi} \sim \mathcal{N}(0,\sigma),$$

with $i_\xi, j_\xi \sim \mathcal{U}(\{1,\ldots,n\})$, $i_\xi \neq j_\xi$ and $(i_{\xi_1}, j_{\xi_1}) \neq (i_{\xi_2}, j_{\xi_2})$, where ξ runs from 1 to l and $\mathcal{U}(A)$ denotes the discrete uniform distribution on the set A.

3.5.5. Fixing the outer covariance matrix

This method transforms a given dataset affinely in a way, such that it follows a given covariance matrix. This is achieved in three steps. In the first step a PCA is performed on the data X, so that eigenvectors are obtained and, scaled by their eigenvalues, stored in a Matrix M_1. Applying M_1^{-1} to X will rotate and whiten the supplied data and yield X'. Next, we perform eigenvalue decomposition of the target covariance matrix Σ and thus obtain a matrix M_2 of the eigenvectors, again scaled by their eigenvalues. We then transform X' by M_2 and yield X_t which satisfies the desired covariance matrix Σ.

Formally, for a data point x, we have

$$x_t = M_2 \cdot (M_1^{-1} \cdot (x - \vec{c})) + \vec{c}.$$

4. Experiments

In this section we demonstrate the usefulness of being able to control characteristic parameters of the classifier evaluation. The WGKS directly controls the number of densities/dimensions, sample size and the Bayes error rate. We use this to exemplarily investigate the question of how the error rate of the 1-nearest-neighbor algorithm (1-NN) truly behaves between the theoretical bounds for k classes, the Bayes error rate e and $e \cdot (2 - \frac{k-1}{k} \cdot e)$ (cf. Duda et al., 2000).

Those bounds however only hold true in the case of an infinite sample size; for a practitioner this knowledge is just a loose guess of what he can expect when applying the nearest neighbor method to real data, which is surely finite and might even be small. It is a priori unclear how the error behaves in this case and in particular, whether it can swell beyond the theoretical bounds. Also, it is viable to know how the error rate changes, when the number of training samples used or the number of classes varies.

For simplification, real-world data is often assumed to be of some kind of Gaussian mixture. Hence the WGKS method with its controlled Bayes error rate is well suited for investigating this problem.

4.1. Experimental setup

We considered datasets with $k = 2, 5, 10, 20$ classes and Bayes error rates e between 0 and $(k-1)/k$ in steps of 0.05 for each k. Also for each k and e training sample sizes of 100, 1000 and 10,000 samples per class density (n) for the 1-NN were considered. The size of the test data set was chosen to be 5000 n. This guarantees that the probability granularity is always the same.

4.2. Experimental results

The experiments were repeated 12 times and results are presented in Figs. 5–8. It can be observed that the behavior of the 1-NN is depending on all three studied parameters (k, e, n).

Three basic observations can be made from the graphs:

1. The variance of the classification rates decreases, when the number of classes k increases.
2. For an increasing k, the average performance curves' sensitivity to n increases. A larger number of samples tends to yield a better performance.
3. The overall performance of the 1-NN classifier decreases with increasing k.

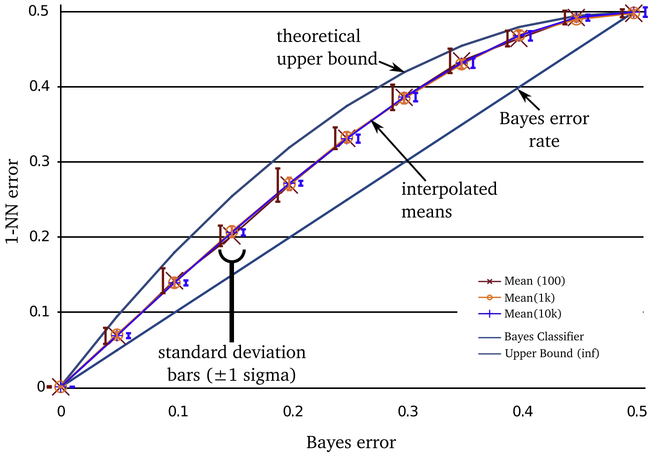

Fig. 5. The average performance curves of the 1-NN classifier on the WGKS data for $k = 2$ classes and $n = 100$ (brown), 1000 (orange) and 10 000 (blue) relative to the Bayes error rate. The vertical bars next to the data points show plus/minus the standard deviation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

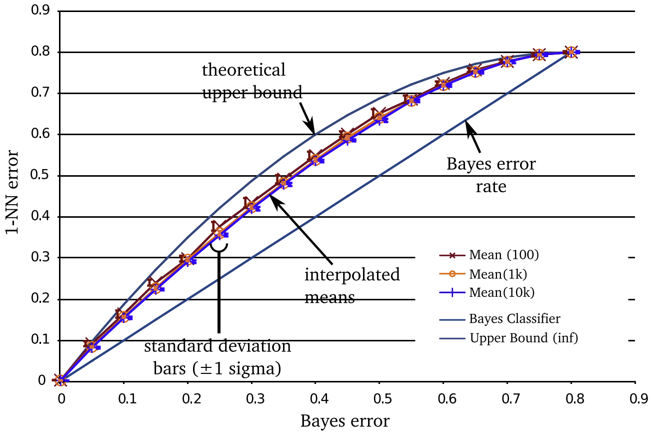

Fig. 6. The average performance curves of the 1-NN classifier on the WGKS data for $k = 5$ classes and $n = 100$ (brown), 1000 (orange) and 10 000 (blue) relative to the Bayes error rate. The vertical bars next to the data points show plus/minus the standard deviation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

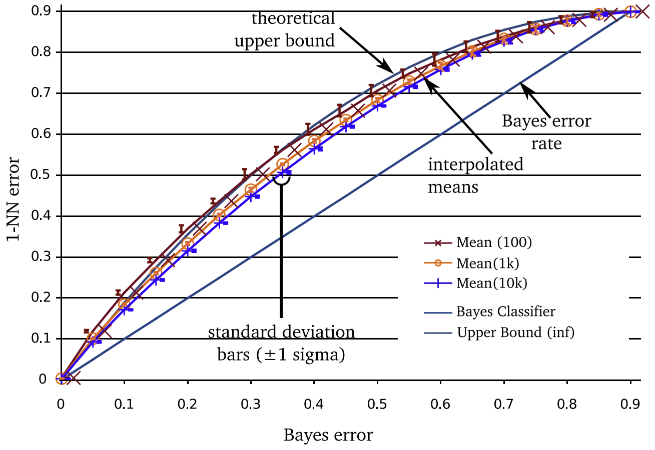

Fig. 7. The average performance curves of the 1-NN classifier on the WGKS data for $k = 10$ classes and $n = 100$ (brown), 1000 (orange) and 10 000 (blue) relative to the Bayes error rate. The vertical bars next to the data points show plus/minus the standard deviation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

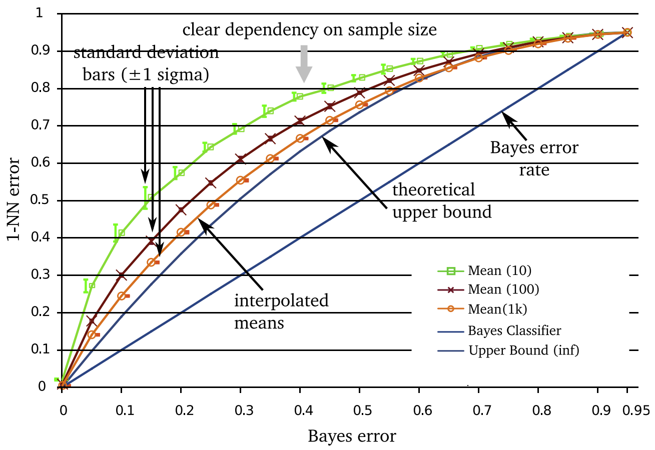

Fig. 8. The average performance curves of the 1-NN classifier on the WGKS data for $k = 20$ classes and $n = 10$ (green), 100 (brown) and 1000 (orange) relative to the Bayes error rate. The vertical bars next to the data points show plus/minus the standard deviation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4.3. Experiment interpretation

With the help of the WGKS data generation model we demonstrated that it is possible to identify the performance distribution of the first nearest neighbor classifier. As the WGKS is an off the shelve tool, we were able to quickly construct an experiment yielding useful information for the practice.

The 1-NN performs best for binary decisions where the first nearest neighbor will have an average performance much better than the theoretical upper bound. Also, for a low number of classes already few samples per density seem to be sufficient for a good performance whereas when k grows, the number of samples per class must be significantly enlarged. In particular, the overall number of samples must grow overproportionally to yield a performance within the theoretical bounds of the infinite case.

5. Discussion

With synthetic data generators it is possible to systematically perform controlled experiments that allow to gain generic insights on a machine learning tool's performance. In contrast, using natural datasets for reference, like the ones found in the UCI repository, limits the study of influence factors (data characteristics), since such characteristics are almost impossible to control independently.

The proposed generator delivers data for testing supervised and unsupervised learning methods on large-scale problems with a high degree of automation. This data can be embedded into high dimensional spaces maintaining controlled statistical properties. In this way our generators can control intrinsic dimensionality. The structure of the data is predictable, repeatable and comprehensive. Our generator delivers true Bayes posteriors that on the one hand allow to compare how well classifiers estimate these and on the other hand define a reference for classifiers' performances.

Concretely, the WGKS generator allows to perform a variety of hypothesis testing experiments by guaranteeing specific stochastic properties. Our generator can guarantee a specific global Bayes error rate, sample size, structure (class relationships), number of dimensions, dataset covariance and data centroid. Apart from that it is possible to add transformations to test for trivial data variations. This is also helpful to find weaknesses in new algorithms as they should not get influenced by these transformations.

A beneficial side effect of using well controlled synthetic data generators in general, and the WGKS in particular, is the simplified reproducibility of scientific work. Instead of actually providing data any author can describe the procedure of data generation, which can be reproduced according to these descriptions. Our implementation of the WGKS produces these descriptions automatically, so that they can be either attached to a publication or made available for download on the internet.

The generator is open source and the Python implementation is available for download at the MADM website.[1] SciPy is required, but beyond this the generators can run almost in any computer environment. SciPy allows exporting of data to many formats including Matlab so that Matlab users can harness the provided generator.

6. Conclusion

Synthetic data generators which can control the statistic properties of the feature vectors are important tools for experimental inquiries performed in context of machine learning and pattern recognition. We proposed a new generator that can be used to model simple problems with fully known statical characteristics. Issues concerning the efficiency of the methods were discussed and significant improvements in speed were presented, making the WGKS method of practical use.

We consider this work as a cue to stimulate further work in this field. In (Lodwich et al., 2009) we demonstrate other synthetic Bayes-true data generators for dynamic processes emulation based on random walks and partially linear models.

Acknowledgments

This work was partially funded by the BMBF (German Federal Ministry of Education and Research), Project PaREn (01 IW 07001).

Appendix A. Variance comparison of Bayes error computation methods

The methods A and B from Section 3.1 for the Bayesian error rate computation will be compared formally in terms of their variance in the following.

We compute the Bayes error rate by $e = 1 - cor$, where *cor* is the fraction of samples being classified correctly by the optimal Bayesian classifier with equal priors. In order to compare the two methods for estimating *cor* we will need the following definitions:

- $\mathbb{1}_M(x)$ is the characteristic function of the set M;
- $c_o(x)$ denotes the ground truth, i.e. the originator class of a sample x;
- $\hat{c}(x)$ denotes the class with the largest Bayesian posterior probability;
- $f(x)$ is the density function of any of the identical densities in the WGKS model;
- $p(M)$ is the probability of a set M;
- A_j denotes the Bayesian acceptance region of density j, A_o denotes the one of the originator class c_o;
- k is number of densities;
- $f^*(x) = \frac{1}{k}\sum_{j=1}^{k} f_j(x)$ is the mixture density of all WGKS densities and
- $|X|$ is the number of samples.

A.1. Method A: confusion matrix evaluation

We have

$$cor = \int \mathbb{1}_{\hat{c}(\cdot)=c_o(\cdot)}(x) \cdot f^*(x)dx \approx \frac{1}{|X|}\sum_{h=1}^{|X|} \mathbb{1}_{\hat{c}(\cdot)=c_o(\cdot)}(x)$$

by standard Monte Carlo estimation, where x is sampled from the mixture of all densities f^*. For the estimator $\mathbb{1}_{\hat{c}(\cdot)=c_o(\cdot)}(x)$ of a sample x we have

$$\mathbb{1}_{\hat{c}(\cdot)=c_o(\cdot)}(x) = \mathbb{1}_{A_o}(x).$$

Hence for the variance of the estimator it holds:

$$var(\mathbb{1}_{A_o}(x)) = p(A_o) \cdot (1 - p(A_o)) = (1 - e) \cdot e.$$

A.2. Method B: importance sampling

Derivation. It holds

$$\begin{aligned} cor &= \int \mathbb{1}_{\hat{c}(\cdot)=c_o(\cdot)}(x) \cdot f^*(x)dx = \frac{1}{k}\sum_{j=1}^{k} \int \mathbb{1}_{A_j}(x) \cdot f_j(x)dx \\ &= \frac{1}{k}\int f_{max}(x)dx, \end{aligned} \tag{2}$$

where $f_{max}(x) = \max_{j=1,\ldots,k} f_j(x)$. This can be approximated, using the idea of importance sampling:

$$\begin{aligned} \frac{1}{k}\int f_{max}(x)dx &= \frac{1}{k}\int \frac{f_{max}}{\frac{1}{k}\sum_{j=1}^{k} f_j(x)} \cdot f^* dx = \int \frac{f_{max}}{\sum_{j=1}^{k} f_j(x)} \cdot f^*(x)dx \\ &= E^*\left(\frac{f_{max}(x)}{\sum\limits_{j=1}^{k} f_j(x)}\right) \approx \frac{1}{|X|}\sum_{h=1}^{|X|} \frac{f_{max}(x_h)}{\sum_{j=1}^{k} f_j(x_h)}. \end{aligned}$$

Note that the x_h are sampled from the joined mixture density.

Variance. We can compute the variance of the estimator $\frac{f_{max}(x)}{\sum f_j(x)}$ by

$$\begin{aligned} var\left(\frac{f_{max}(x)}{\sum f_j(x)}\right) &= E^*\left(\left(\frac{f_{max}(x)}{\sum f_j(x)}\right)^2\right) - \left(E^*\left(\frac{f_{max}(x)}{\sum f_j(x)}\right)\right)^2 \\ &= \frac{1}{k}\int \frac{(f_{max}(x))^2}{(\sum f_j(x))^2} \cdot \sum_{i=1}^{k} f_i(x)dx \\ &\quad - \left(\frac{1}{k}\int \frac{f_{max}}{\sum f_j(x)} \cdot \sum_{i=1}^{k} f_i(x)dx\right)^2 \\ &= \frac{1}{k}\int \underbrace{\frac{f_{max}(x)}{\sum f_j(x)}}_{\leqslant 1} \cdot f_{max}(x)dx - \left(\underbrace{\frac{1}{k}\int f_{max}(x)dx}_{=1-e by\ (2)}\right)^2 \end{aligned}$$

since the densities are identical and have the same error rate. This can further be reduced to

$$var\left(\frac{f_{max}(x)}{\sum f_j(x)}\right) \leqslant (1 - e) - (1 - e)^2 = (1 - e) \cdot e = var(\mathbb{1}_A(x)),$$

the variance of the confusion matrix estimator.

Hence it holds *var*(estimator (A)) ⩾ *var*(estimator (B)) and it is easy to see that estimator (B) is better in the case of $f_{max}(x) < \sum_{j=1}^{n} f_j(x)$.

[1] https://madm.dfki.de/downloads.

Appendix B. Transformation for Bayes error computation speed-up

B.1. Formal verification of results

Lemma 1. *Let $D_1, D_2, \ldots, D_k$ be $k = d + 1$ densities with centers $\mu_1, \mu_2, \ldots, \mu_k$ located on the corners of a regular k-simplex in an d dimensional Euclidean Space. Let $B_{j,l}$ be the $d - 1$ dimensional perpendicular line bisector of the centers of the densities D_j and D_l, $j \neq l$. If*

(a) *all pairs of densities D_j and D_l are symmetric by reflection through $B_{j,l}$ and*
(b) *each density D_i is mirror symmetric by reflection through each $B_{j,l}$ $\forall j,l \in \{1,\ldots,k\}\backslash\{i\}$, $j \neq l$,*

then all densities are positioned rotationally invariant, i.e. a rotation by any true rotation $S \in$ Sym(k-simplex), the symmetrical group of the k-simplex, does not change the overall setup except for the labels of the densities in the respective global position.

Proof. From elementary geometry we know that in 2D a rotation is equivalent to two concatenated reflections through lines α and β (Kane, 2001). The angle of the rotation equals two times the angle between α and β. This can be generalized to 3D (Beardon, 2005) and arbitrary dimensions (Kane, 2001), using dihedral angles (the angles between two hyperplanes). Hence we can express a rotation $S \in$ Sym(k-simplex) by a concatenation of an even number of reflections through some hyperplanes $B_{\cdot,\cdot}$. Fomin and Reading (2008) show that these hyperplanes indeed are the perpendicular line bisectors. It is also shown there that due to the setup of the densities on a regular k-simplex each $B_{j,l}$ runs through all other centers x_i, $i \neq j$, l, hence fixes them. By assumption (b), for each reflection through $B_{j,l}$, all other densities X_i remain in an equivalent state. By assumption (a) the densities X_j and X_l are symmetric by reflection through $B_{j,l}$, hence for the overall setup this just results in a swap of the labels j and l. All in all this means the simplex is rotation invariant to any rotation from Sym(k-simplex). □

Theorem 1. *Let $N_1, N_2, \ldots, N_k$ be $k = d + 1$ identical Normal distributions with covariance matrices $\Sigma_i = \sigma^2 I$ and centers μ_i, $i = 1,\ldots,k$ located on the corners of a regular k-simplex in an d-dimensional Euclidean Space E. Let $B_{i,j}$ be the $(d - 1)$-dimensional perpendicular line bisector of the centers of the densities N_i and N_j, $i \neq j$. Then $B_{i,j}$ are the Bayesian decision boundaries. Furthermore, the overall Bayesian error rate is equivalent to the error rate of each density.*

Proof. $B_{i,j}$ is the Hyperplane consisting of all equidistant points between μ_i and μ_j. Since N_i and N_j are identical and have diagonal covariance matrices (being scalar multiples of the identity matrix), the Bayesian decision boundary (assuming equal priors) is $B_{i,j}$ (Duda et al., 2000). The boundaries $B_{i,j}$ $\forall i,j = 1,\ldots,k$, $i \neq j$ subdivide E into k compartments, each defined by $\{B_{i,j} | j \in \{1,\ldots,k\}\backslash\{i\}\}$ corresponding to the space where one density N_i dominates the others. Obviously, a pair of densities N_j and N_l are symmetric by reflection through $B_{j,l}$ and clearly each density N_i is invariant to reflection through any other $B_{l,j}$, $l,j \in \{1,\ldots,k\}\backslash\{i\}$, $l \neq j$, since such a $B_{l,j}$ runs through μ_i. By Lemma 1 this means the densities arranged on the simplex are rotational invariant and hence the error rate in each compartment must be the same and hence be equivalent to the overall error rate. Again as a result from the symmetry of N_i and N_j (and the invariance of all other densities), the density mass that is misclassified as "Class i" though belonging to N_j equals the density mass that is misclassified as "Class j" though belonging to N_i. Using this argument for all densities N_j neighboring N_i, we get that the error rate in each compartment is equivalent to the error rate of each single density N_i. □

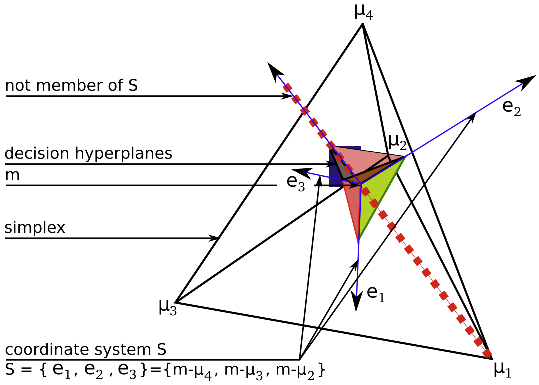

Fig. 9. Obtaining of the coordinate system *S*.

B.2. Detailed description of the transformation

In detail, the transformation described in Section 3.2 works as follows. Note that all vector coordinates are expressed in terms of a global coordinate system where N_1 is located at the origin and N_2 is located on the x_1-axis.

1. Let the k-simplex be centered at its centroid. Let S (Fig. 9) be the coordinate system consisting of the direction vectors of the equidistance lines of each $(k - 1)$ density centers (with one thereof always being μ_1, the center of N_1) in an order such that μ_1 lies in the first orthant. The basis vectors e_i spanning S can be computed by $e_i = m - \mu_{k-i}$ $\forall i = 1,\ldots,k - 1$, where m is the centroid of the simplex.
2. Transform the coordinate system to a Cartesian one S'. Compute μ'_1 by $\mu'_1 = S^{-1} \cdot \mu_1$.
3. Since S^{-1} is a linear transformation, to ensure equivalence of the classification problems, it suffices to sample from a normal density $N'_1 = \mathcal{N}(\mu'_1, \Sigma')$ in the new coordinate system, where $\Sigma' = S^{-1} \cdot \Sigma \cdot (S^{-1})^T = S^{-1} \cdot (S^{-1})^T$ if the original Σ was the unit matrix.

References

Abowd, J.M., Lane, J., 2003. Synthetic Data and Confidentiality Protection. Technical Papers, Longitudinal Employer-Household Dynamics, Center for Economic Studies, US Census Bureau. <http://econpapers.repec.org/RePEc:cen:tpaper:2003-10>.

Baird, H.S., 1993. Document image defect models and their uses. In: Proc. Second Internat. Conf. on Document Analysis and Recognition ICDAR-93, pp. 62–67.

Baird, H.S., 2000. The state of the art of document image degradation modeling. In: Proc. 4 th IAPR Internat. Workshop on Document Analysis Systems, Rio de Janeiro, pp. 1–16.

Beardon, A.F., 2005. Algebra and Geometry. Cambridge University Press.

Csallner, A.E., Csendes, T., Csaba Markót, M., 2000. Multisection in interval branch-and-bound methods for global optimization – I. Theoretical results. J. Global Optimiz. 16 (4), 371–392.

Davidov, D., Gabrilovich, E., Markovitch, S., 2004. Parameterized generation of labeled datasets for text categorization based on a hierarchical directory. In: SIGIR '04: Proc. 27th Annual Internat. ACM SIGIR Conf. on Research and Development in Information Retrieval. ACM, New York, NY, USA, pp. 250–257.

Duda, R.O., Hart, P.E., Stork, D.G., 2000. Pattern Classification. Wiley-Interscience Publication.

Fomin, S., Reading, N., 2008. Root Systems and Generalized Associahedra. http://arxiv.org/abs/math/0505518.

Genz, A., 1999. Methods for Generating Random Orthogonal Matrices. Monte Carlo and Quasi-Monte Carlo Methods 1998. Springer-Verlag, Berlin.

Helmers, M., Bunke, H., 2003. Generation and use of synthetic training data in cursive handwriting recognition. In: IbPRIA, pp. 336–345.

Jeske, D.R., Samadi, B., Lin, P.J., Ye, L., Cox, S., Xiao, R., Younglove, T., Ly, M., Holt, D., Rich, R., 2005. Generation of synthetic data sets for evaluating the accuracy of knowledge discovery systems. In: KDD '05: Proceedings of the Eleventh ACM

SIGKDD International Conference on Knowledge Discovery in Data Mining. ACM, New York, NY, USA, pp. 756–762.

Kane, R., 2001. Reflection groups and invariant theory. In: CMS Books in Mathematics. Springer-Verlag.

King, R.D., Feng, C., Sutherland, A., 1995. Statlog: comparison of classification algorithms on large real-world problems. Appl. Artif. Intell. 9 (3), 289–333.

Limpert, E., Stahel, W.A., Abbt, M., 2001. Log-normal distributions across the sciences: keys and clues. BioScience 51 (5), 341–352.

Lodwich, A., Frasch, J., Breuel, T.M., 2009. Practical Bayes-true Data Generators for Evaluation of Machine Learning, Pattern Recognition and Data Mining Methods. Tech. Rep. German Research Center for Artificial Intelligence (DFKI).

Pei, Y., Zaïane, O., 2006. A Synthetic Data Generator for Clustering and Outlier Analysis. Technical Report.

Rachkovskij, D.A., Kussul, E.M., 1998. Datagen: a generator of datasets for evaluation of classification algorithms. Pattern Recognition Lett. 19 (7), 537–544.

Rogers, M., Graham, J., Tonge, R.P., 2003. Using statistical image models for objective evaluation of spot detection in two-dimensional gels. Protemics 3 (6), 879–886.

Srikant, R., 1999. IBM quest synthetic data generation code (not available online anymore), not available online anymore. <http://www.almaden.ibm.com/software/quest/Resources/datasets/syndata.html>

Theodoridis, Y., Silva, J.R.O., Nascimento, M.A., 1999. On the generation of spatiotemporal datasets. Lecture Notes in Computer Science, vol. 1651. Springer, pp. 147–164.

UCI, 2009. University of california/irvine – machine learning repository.

van der Walt, C.M., Bernard, E., 2007. Data characteristics that determine classifier performance. SAIEE Africa Res. J. 98 (3), 87–93.

Wolpert, D.H., Macready, W.G., 1997. No free lunch theorems for optimization. IEEE Trans. Evol. Comput. 1 (1), 67–82.