MLPM project
The effect of smoothness

Maarten van der Velden
5743087
Maarten.vanderVelden@student.uva.nl

Carsten van Weelden
0518824
cweelden@science.uva.nl

November 9, 2011

1 Introduction

In this paper we investigate the effect of smoothness of the dataset on the performance of classification algorithms. In order to investigate this we generate several artificial datasets of varying smoothness and look at the accuracy and loss of the resulting classifiers. In order to get insight into the effects we decompose the loss into its bias, variance, and noise components as defined in [**?**].

To investigate the effect that smoothness has on classification performance we first need to define some idea of smoothness which we can easily vary and then run a set of experiments for different levels of smoothness. We view classification as a two class problem[1], with each class being represented by some Probability Density Function (PDF) over the attribute space. Given this view the *smoothness* of a class is determined by the shape of the PDF. We define the PDF for each class as a Gaussian Mixture Model (GMM) with k mixture components. Intuitively, the smoothness is then determined by the number of components in the mixture. With just one component the PDF corresponds to a Gaussian distribution which is very smooth, while increasing the number of components increases the peakedness of the distribution, making it less smooth.

One way to see this is in terms of inherent noise in the problem, which we define as the noise in definition 4 of [**?**]:

$$N(x) = E_t[L(t, y_*)] \tag{1}$$

If we keep the mean variance of the distributions equal, then with more components we will have a higher average probability that the PDFs overlap for a datapoint x. Since the Bayes optimal prediction predicts the class for which the PDF is highest at x, the noise is proportional to the area under the PDF

[1] As contrasted with a concept learning problem, in which there is one class which needs to be distinguished from the background.

for which the PDF of the other class is higher. In other words, the more components make up the distribution of a class, the more overlap there is between classes, and therefore the higher the noise component in the loss is.

To show this effect we generated GMMs with an increasing amount of components and measured their noise. Averaged over 25 random models of the same number of components, it is clear that the noise increases with the number of components, growing asymptotically towards 0.5, which represents the amount of noise where the Bayes optimal decision is correct half of the time: chance level. The results are shown in Figure 1.

2 Experimental method

1. We generate a problem containing two classes denoted C_0 and C_1. Each class is represented by a PDF which we define as a GMM with k components[2]. We do this for $k = [1..5]$. We pick the parameters corresponding to component j of class i as follows. The prior probability for each component is $\pi_{ij} = rand([1..5])/Z_i$ where Z_i is a normalizing factor such that $\Sigma_j pi_{ij} = 1$. The mean of each component is $\mu_{ij} = (rand([-1, 1]), rand([-1, 1]))$. We use a scalar covariance matrix σI with $\sigma = rand([0.1, 0.4])$.

2. For each two class problem we generate a set of 100 training sets $D = \{d_1, d_2, ..., d_{100}\}$ and a corresponding test set T. We do this for $|D| \in 10, 100, 1000, 1000$ with half of the data points being sampled from each class. For the test set $|T| = 1000$, also with half of the points being sampled from each class.

3. We train a classifier on each training set $d \in D$ and evaluate the accuracy on the test set.

4. Finally, we compute the average bias and average variance on T.

3 Results

We trained Naive Bayes classifiers and kNN classifiers for all settings of the GMM we defined in the last section. The resulting error is shown in Figure 2. It can be seen that the loss is dependent on the amount of training data, with larger numbers of data lowering the loss (this improvement seems to slow towards more data being usd). Furthermore, kNN seems to work better for low amounts of training data. The difference between NB and kNN with more data is small, but kNN tends to be better with less smooth data and NB with smoother data.

Furthermore, the loss depends on the smoothness of the data: the more GMM components used per class, the higher the loss. The results show crudely the same trend as does the amount of noise with increased smoothness, so it might be the case that the increased loss is caused mainly by an increased amount of noise. Therefore, the bias and variance components of the loss were estimated separately. these results can be seen in Figures 3 and 4. From Figure 3, it is clear that the bias increases sharply when the smoothness of the data is reduced. There is little difference in the amounts of data used here, and again

[2]We use a 2-dimensional attribute space for ease of visualization.

kNN seems to have more bias than NB on smooth data and less bias on less regular data. The variance component of the error does also increase with less smooth data, but less sharply than the bias does. Here, it becomes clear that the fact that training on small amounts of data performs less can be attributed to the higher variance, because the noise and bias don't differ significantly over this variable. It is clear too that kNN on low amounts of data is better due to a smaller variance component in the loss. For the other amounts of training data, NB seems to have consistently a little less variance.

We were interested in the relation between the loss of the classifiers and the components bias, variance and noise. As was reported in several papers (*cf.* [**?**]), this relationship can be of a complex nature, although it is generally assumed to be a sum. To find a hint about this relationship in the algorithms used, we compared the loss with the naive way of combining the components: summing. The results are shown in Figures 5 and 6.

The results show that when normalized, the loss of each classification setting is very similar to the sum of its components, which suggests a linear relationship between them. It is not a simple sum because the sum of the components tends to be larger than the loss, but the graphs suggest there is no real intricate relationship with the settings used and the classifiers used.

4 Conclusion

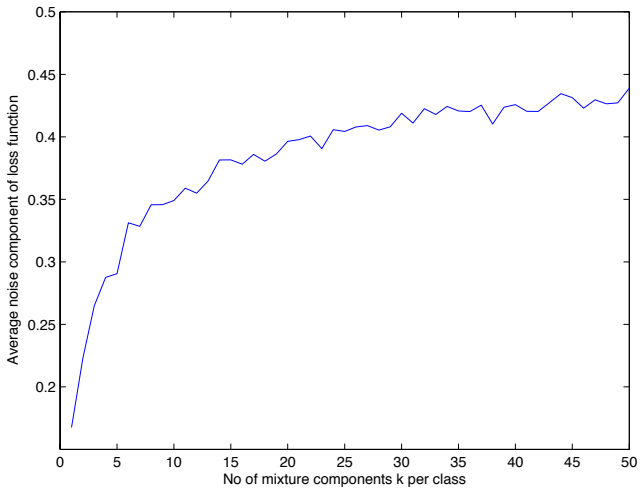

Figure 1: The effect on the noise of the use of more mixture components in the GMM.

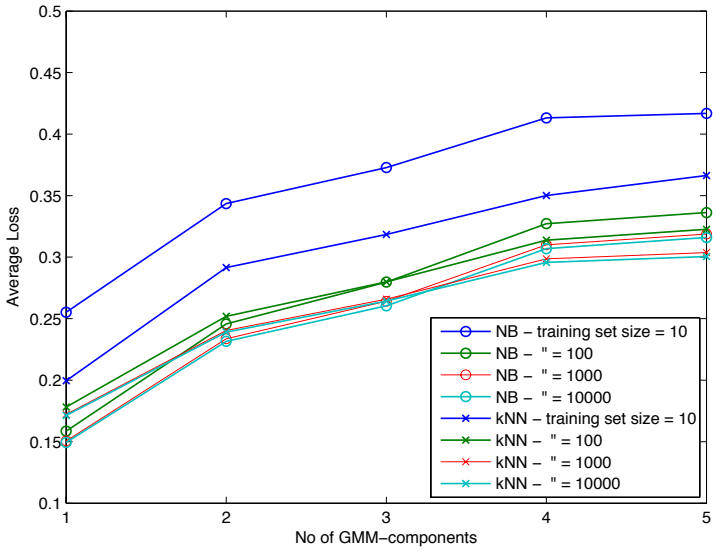

Figure 2: The average error (loss) of the classifiers on different amounts of training data, given different levels of smoothness of the data.

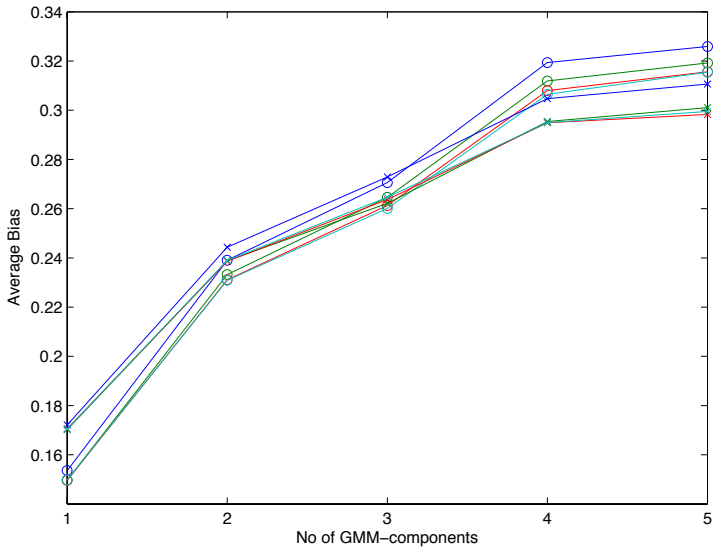

Figure 3: The average bias of the classifiers on different amounts of training data, given different levels of smoothness of the data.

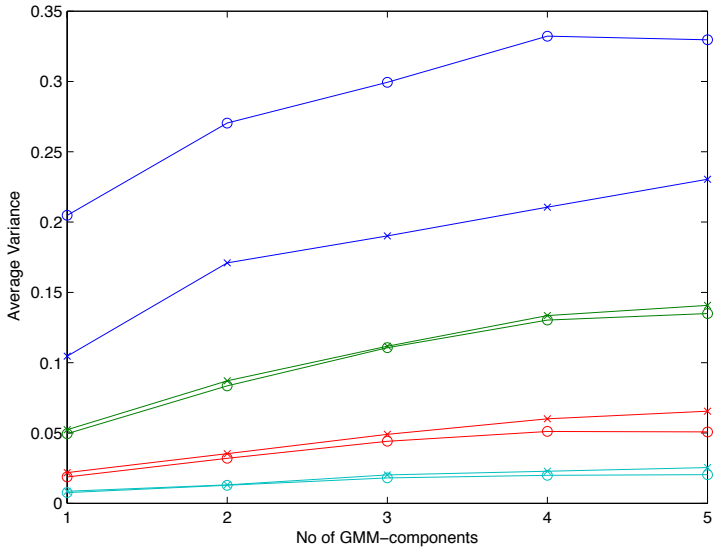

Figure 4: The average variance of the classifiers on different amounts of training data, given different levels of smoothness of the data.

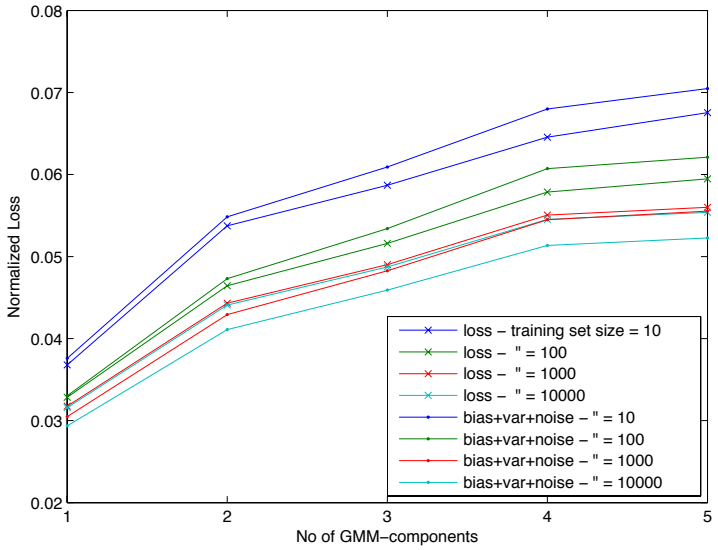

Figure 5: KNN. Comparison of the average loss with the sum of the average bias, variance and noise. Each is normalized so all values for a data series sum to 1.

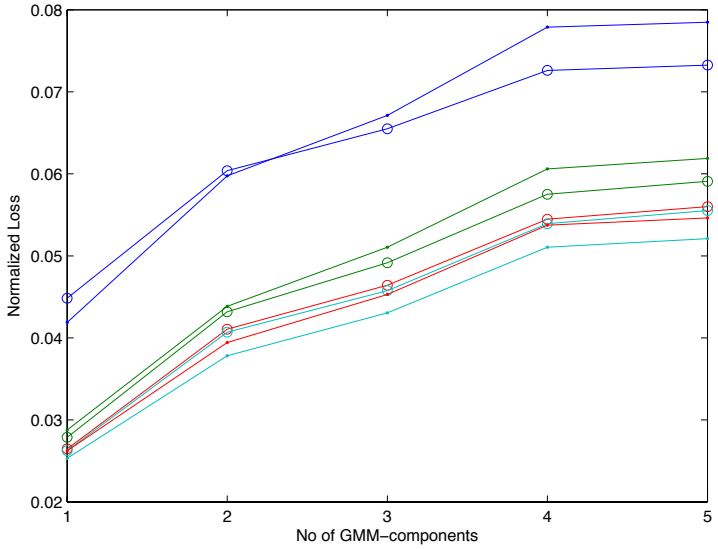

Figure 6: Naive Bayes. Comparison of the average loss with the sum of the average bias, variance and noise. Each is normalized so all values for a data series sum to 1.