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 standard deviation of the distributions approsimately 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

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

the PDF 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. The models where generated according tot he method described in section 2. 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.

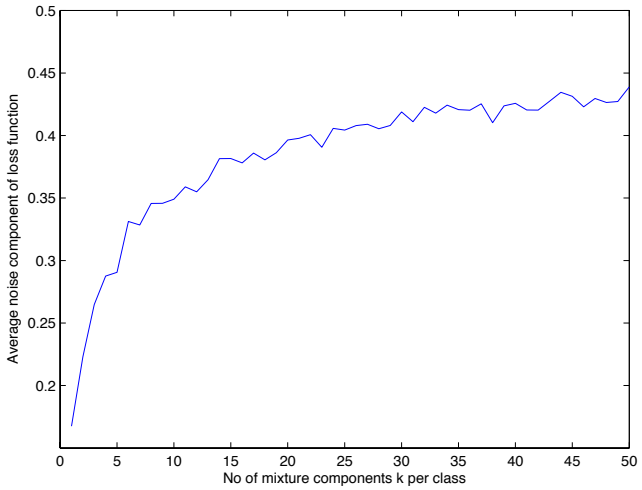

Figure 1: The effect on the noise of the use of more mixture components in the GMM.

2 Experimental method

To investigate the effect that the smoothness of the distributions has on classification performance we run a set of experiments. We generate a set of two-class problems with a given smoothness (as characterized by the number of components in the GMM representing each class) and for each problem we estimate the expected loss, average bias, and average variance.

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

2. For each two class problem we generate a set of 50 training sets $D = \{d_1, d_2, ..., d_{50}\}$ 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 loss on the test set.

4. Finally, we compute the average bias and average variance on T over all the training sets $d \in D$.

We compute the loss as 0/1-loss and report the mean over all training sets and all problems for each number of mixture components and for each size of training set. We compute the average bias and average variance as given in [**?**] and again report the mean over all training sets and all problems given the number of components in each class and the training set size.

We perform these experiments with two different classifiers: Naive Bayes (NB) and k-Nearest Neighbour (KNN). NB does not have any parameters, so we simply train it once and calculate its predictions. For KNN we have to select K, the number of neighbours. Since we are interested in the best KNN classifier possible for each problem and training set, we perform an oracle run and select the K which gives the best result on the test set. We do not try every value for K but we sample 10 equidistant values from the range $[1..|D|]$, thus ranging from Nearest Neighbour classification to simply predicting the mean over the whole data.

We expect that for less smooth problems the expected loss will be higher, the intuition being that less smooth problems are harder to learn. For the same reason we expect the bias component of the loss to increase with the number of components per class. For both we expect them to be lower for larger training sets. For the average variance we expect to see that it increases with the amount of mixture components, but we expect this increase to be greater for smaller sizes of training set, since there are less datapoints per mixture component.

3 Results

3.1 KNN

In contrast to training a Naive Bayes classifier, for KNN it is important to choose a sensible value for K, the number of neighbors taken into account. Because training KNN completely for some different values of K only to find out good ones is very expensive, we try a range of K as described in the previous section, classify for each of these values and store the highest performance. To be able to see what this leads to, we put the occurrences of different K's in histograms (Figures 2 and 3 show the results for a training set of 10 samples and 10000 samples respectively).

The histograms show a preference for relatively small values of K, especially for larger training set sizes. This however might also be caused by the speherical Gaussian nature of the distribution of the data. The probability for each class decreases proportionally to the distance to the mean. It is likely that a small

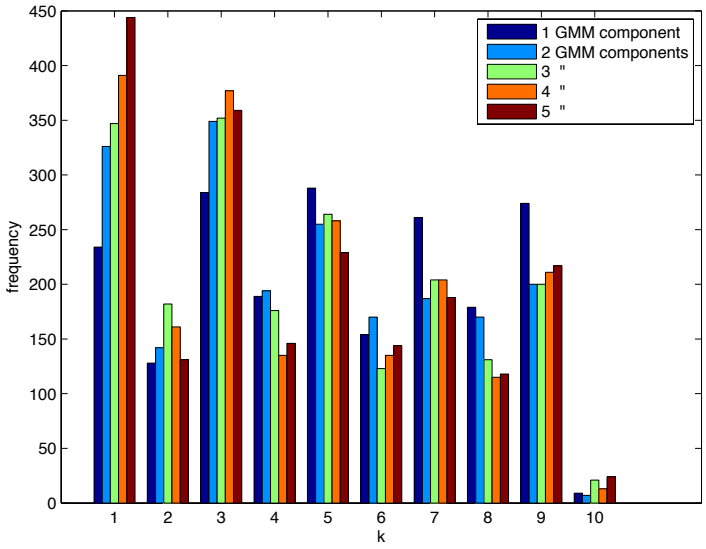

Figure 2: Histogram of the amount of occurrences of different values of K in KNN on a training set size of 10, over all trained classifiers, for different levels of smoothness (No of GMM components).

amount of data gives a quite good estimation of the mean of the distribution, therefore one tenth of the training set size might be quite big already as the training set gets bigger itself.

3.2 Classification

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 4. 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 used). 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 5 and 6. From Figure 5, 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 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

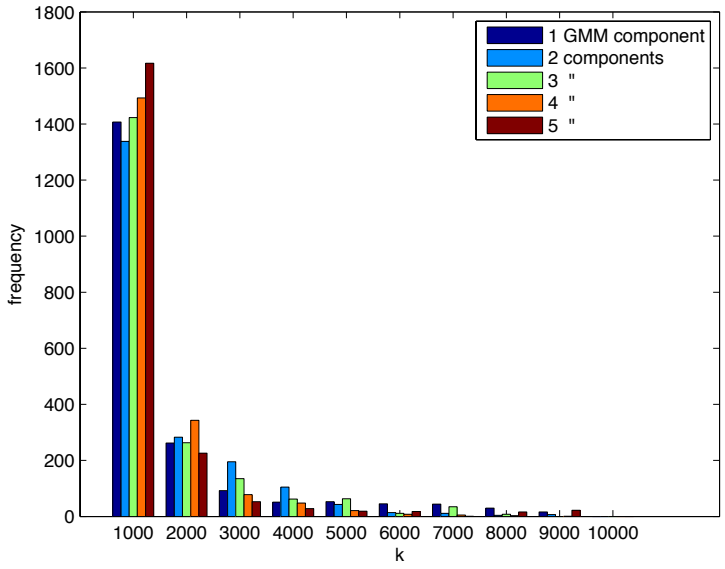

Figure 3: Histogram of the amount of occurrences of different values of K in KNN on a training set size of 10000, over all trained classifiers, for different levels of smoothness (No of GMM components).

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 7 and 8.

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 the relationship is approximately proportional with the settings used and the classifiers used.

4 Conclusion

From these result we can draw several conclussions. For sufficiently large training sets, KNN performs better[3] than NB when the data is less smooth, while NB performs better on smoother data. This seems to be because for less smooth data the bias component of the loss is smaller for KNN while for smoother data

[3]as measured by average loss

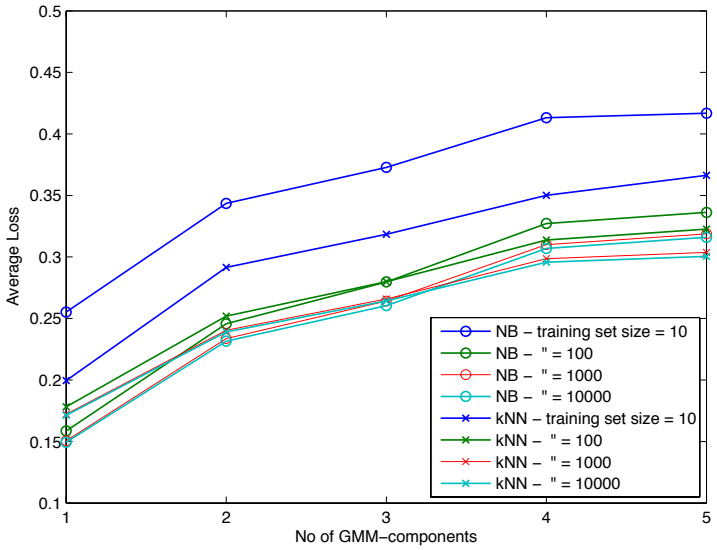

Figure 4: The average error (loss) of the classifiers on different amounts of training data, given different levels of smoothness of the data.

it is smaller for NB. This can be explained by the fact that NB generates relatively smooth distributions while KNN seems to have a preference for relatively small values of K for these type of Guassian classification problems.

Furthermore, for very small small datasets KNN consistently outperforms NB for all levels of smoothness. Even though the mean average bias is larger for smooth data, the mean average variance is much lower. This effect is not seen for larger datasets in which case the average variance is roughly the same, or a bit higher for KNN. With such a small dataset, NB does not have enough data to correctly estimate the conditional probabilities, but the Nearest Neighbour method does not have this problem.

In general the experiments show that average loss decreases for smoother problems and although the loss is lower for larger datasets, the decrease relative to the smoothness behaves the same over different sizes of dataset. This decrease seems to be mainly from lower bias for smooth problems, showing that the learning methods have less trouble approximating the smoother functions. The variance also decreases for smoother problems, but this decrease is less marked and seems to be less for larger datasets. This shows that for KNN and NB the variance in learning is determined more by the size of the dataset than the difficulty of the classification problem.

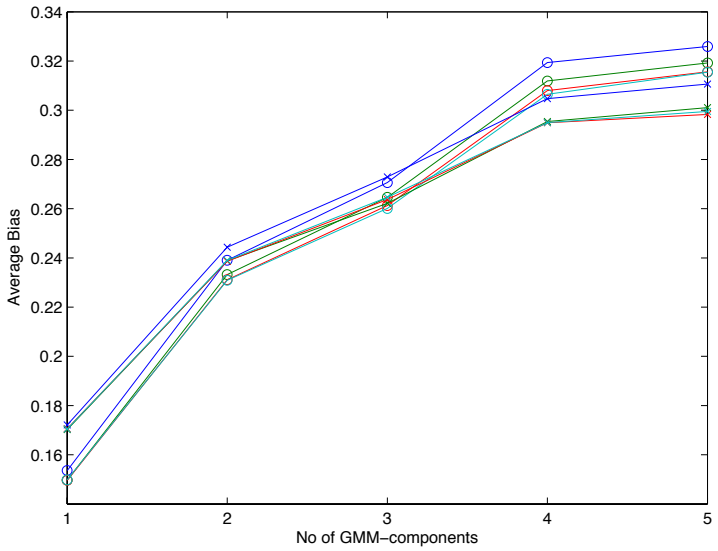

Figure 5: The average bias of the classifiers on different amounts of training data, given different levels of smoothness of the data.

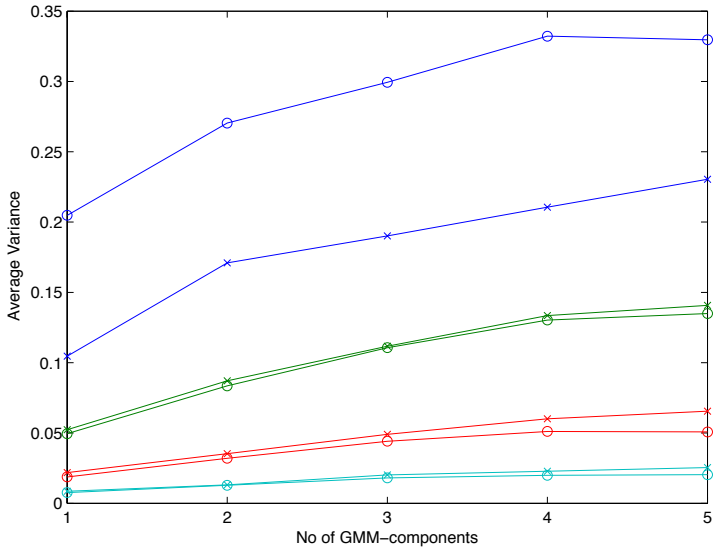

Figure 6: The average variance of the classifiers on different amounts of training data, given different levels of smoothness of the data.

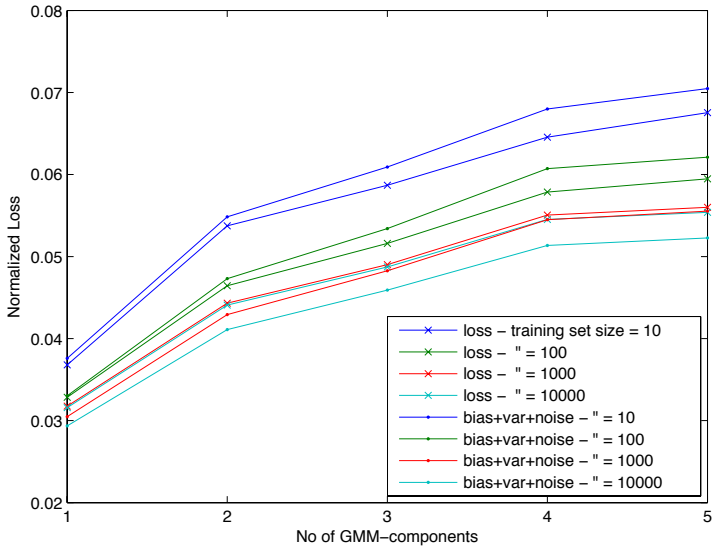

Figure 7: KNN. Comparison of the average loss with the sum of the average bias, variance and noise. Each is normalized so all values for a data series sum to 1.

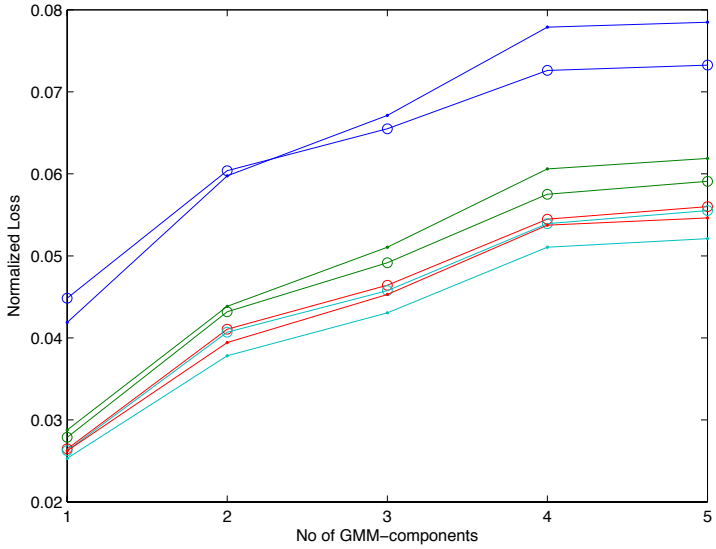

Figure 8: Naive Bayes. Comparison of the average loss with the sum of the average bias, variance and noise. Each is normalized so all values for a data series sum to 1.