Dynamic Connect-4
Assignment 1
Artificial Intelligence - ECSE526

Otis Samuel

October 17, 2017

Instructor: Jeremy Cooperstock

1 Introduction

This report shows the implementation of two classifiers. Using both Gaussian Naive Bayes (GNB) and k-Nearest-Neighbors (kNN) classifiers on featurized music tracks, this report tries to show the limits and benefits for each of them. Each of the songs have been featurized with a frequency analysis of short segments belonging to the whole song. That is, a song contains multiple vectors of 12-dimensional features. Moreover, each song has an attributed genre. Using the classifiers, we try to discover the music genres of a testing set by training our classifiers on a training set of approximately 1500 songs. First, using a GNB classifier, the mean vector and covariance matrix of each genres are extracted. This becomes the model for our Naive Bayes classifier. We can then estimate the probability that a song belongs to the gaussian distribution of each genre. Secondly, using a kNN classifier, no training is in fact required. The classifier simply keeps memory of all its training set and then tries to estimate the class of a song by looking at the euclidean distance between the song features and its k-nearest neighbors' features. The neighbors classes being known, we can estimate the class of the unknown song.

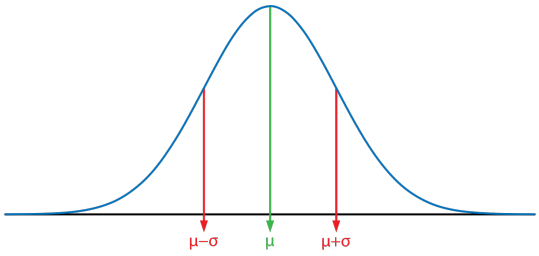

Figure 1: Gaussian (normal) distribution

2 GNB Classifier

2.1 Assumptions

It is important to note, before anything, that a Naive Bayes (NB) classifier assumes that the attributes (for instance, the features) for every class are independant from each other. As stated in [4] and [3], an NB follows Bayes' theorem :

$$P(C_i|E) = \frac{P(C_i)P(E|C_i)}{P(E)} \tag{1}$$

meaning the probability of E, the attributes' space, belonging to class C. From equation 1, it is possible to get the classifier under the form of

$$f_i(E) = P(C_i) \prod_{j=1}^{a} P(A_j = v_{jk}|C_i) \tag{2}$$

Here, f, an NB classifier, assumes that all attributes are independant given a class C. These classifiers are called naive because they assume independance among the attributes, which is often wrong in reality where multiple features may depend from each other. A second assumption made by an NB is that each of the features contribute equally to the outcome, which us often false. For instance, when trying to classify brain electroencephalograms, we will often give more importance to a particular frequency such as the delta frequency band (0.5 - 4 Hz) to detect sleeping states.
We obtain an GNB classifier when we assume that each attribute follows a gaussian distribution, as per figure 1. This third assumption considers that each attributes has its own mean and variance values, fulfilling the probability density of a gaussian distribution :

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{3}$$

However, the data we are using in this project, for example, does not have a gaussian distribution. In fact, it has been proven by [1] that music genres follow a laplacian distribution. But for the sake of simplicity, this project uses a gaussian distribution, since they can be quite similar.

2.2 GNB performances

An NB will mostly work best when using a huge dataset. Moreover, it is faster than most machine learning algorithms and often outperforms more sophisticated ones. Its assumption are that features are independant. However, a simple NB is used only for discrete data. In this project, we use data with continuous values, that is, the frequency analysis. We then need to assume a numerical distribution of the data, that is, a gaussian distribution in this case.
In a case where data is assumed to be independant and having a gaussian distribution, the GNB classifier will work well. However, if we introduce too much prior probabilities in the model, it is easily subject to overfitting. Also if the data has a totally different distribution, its accurracy will be decreased.
In other cases where data is strongly proven to be dependant, naive bayes will often lead to wrong answers. Another classifier which works well with highly dependant attributes is the Maximum Entropy Classifier (MEC). This one is used mostly when we don't know much about the prior distribution of the data. It would perform well in natural language processing, where words are dependant from each other. In the case of spam filtering, NB is still good because it only searches for keywords and not the dependance between them.
Even if is possible to model continuous data with a GNB, its performances may still be poor if the data doesn't follow well the prior gaussian distribution we assume it has. In this work, a basic GNB showed 28% accuracy results on the testing set, which is really poor compared to the kNN classifier, which could go up to 52%. This is due to the kNN being non-parametric. Because it has no prior assumption over the dataset, it leads to better results on many datasets.
On the other hand, if computation speed is an issue, GNB is to be chosen since its training and classifying speed are fast and simple. An kNN classifier has poor time performance the more we have datapoints. In this project, kNN has terrible time efficiency because we have multiple dimensions and a huge datasets of points. GNB works works well with multiple dimensions since it splits the work of multidimensional classifying in many one-dimensional problems. The kNN has to compute euclidean distances on 12 dimensions, which brings in the curse of dimensionality. Many dimensions increases computation times dramatically. To avoid this it is possible to use some prior preprocessing on the data like Principal Component Analysis (PCA) to reduce dimensionality of features in a way to keep maximum variance of the data.

3 kNN Classifier

The only parameter to evaluate, ironically with an non-parametric kNN classifier, is the k number of neighbors. This type of classifier will use k closest neighbors in euclidean distance to assign a class to the feature being compared to the memory of the classifier. We usually take for granted that k is an odd number to avoid tie situation. It is important to know that k acts as a smoother in the classifier. The higher the value of k, the more smoothing will take place.

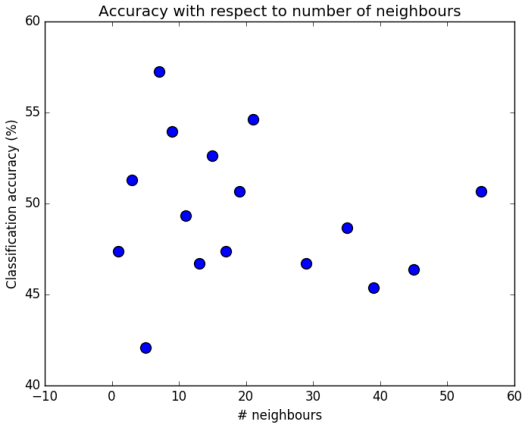

Figure 2: Accuracy of k neighbours

At some point, a too high k value will underfit the data. Normally, n-fold cross-validation is used to determine the optimal value of by looking at classification accuracy results. A thumb rule also exists where k is the square root of the number of data points. For instance, a good value of k for a dataset of 1500 songs would be around 39. In figure 2, we look at the accuracy of one-fold classification result with increasing value of k. However, it is possible to see that the accuracy increases and maxes out around 9-11 neighbours but eventually goes down again because of underfitting. With a 10-fold cross-validation, it would be possible to get smoother accuracy results, but the algortihm is too time-consuming for this.

4 GNB vs kNN

In the end, GNB gave poor classification results of around 28% but with great computation speed. On the other hand, kNN had better results with an accuracy of 52% with a weighting gaussian kernel. By giving a gaussian varying importance for neighbors of the query point in the classification, the accuray went from 50% to 52%. However, the computation time is tremendously sad when using kNN. A given kNN tool already existing, such as sk-learn or tensorflow, could yield better results, but the aim of this project was to program these classifiers ourselves. It is important to note that, in machine learning, the quality of the features is as important as the classifier, if not more. In thi case, we had the frequency analysis of the songs, but a better approach, as described in [2], would be to use wavelet package transform (WPT) and the best basis algorithm (BBA) which functions like a binary search tree in which we choose the wavelet package with the least shannon entropy to describe the signal. Thereafter, a support vector machine (SVM) was used to classify the

data in pair-wise comparisons. In [2], the feature extraction is more important. If the same method was used to extract feature from the dataset we were given, it would be possible to achieve much higher results.
In the end, kNN is better than GNB because it does not make the false assumption that the music points are independant from each other and that they contribute equally to the outcome. All the prior assumptions of GNB, including gaussian distribution of the data, contribute to its downfall in this particular context.
An interesting alternative would be to use a recurrent neural network (RNN). They are known to have dynamic temporal behavior. They would then be able to keep track of inter-feature dependance as well as time dependance, which has a big weight in guessing music genre. Plus, an RNN would be really fast at classifying music genres for the same assignment, thus outclassing all studied classifiers in this work.

References

[1] V. Arora and R. Kumar. Probability distribution estimation of music signals in time and frequency domains. In *2014 19th International Conference on Digital Signal Processing*, pages 409–414, August 2014.

[2] S. H. Chen, S. H. Chen, and T. K. Truong. Automatic music genre classification based on wavelet package transform and best basis algorithm. In *2012 IEEE International Symposium on Circuits and Systems*, pages 3202–3205, May 2012.

[3] Pedro Domingos and Michael Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine learning*, 29(2):103–130, 1997.

[4] Harry Zhang. The optimality of naive Bayes. *AA*, 1(2):3, 2004.