ECSE 526
Assignment 2
Music Genre Classification

Sean Stappas
260639512

October 19th, 2017

Introduction

Gaussian and k-nearest neighbour (kNN) classifiers were created to classify songs into ten distinct genres. The `numpy` and `pandas` libraries were used to easily read and manipulate the test and training feature vectors from CSV files. The `matplotlib` library was used to create all the plots. To improve performance for the online Kaggle competition, multiple elements were used from the `scikit-learn` Python library. These include using their implementation of a k-d tree for kNN, as well as using many of their other classifiers, like the SVM, Naive Bayes, Multi-layer Perceptron, ADA, QDA and Gaussian Process classifiers. Out of all these classifiers, the kNN was found to be best. The focus of this report, however, will be to compare the simple Gaussian and kNN classifiers.

1 In general, what assumptions about the data do we make when we model the data using a Gaussian distribution?

Fundamentally, when we model the data using a Gaussian distribution, we assume that it has a "Gaussian" shape. In the univariate case, this means that the distribution has the familiar "bell curve" shape. In the multivariate (n-dimensional) case, however, the contours of the distribution form an n-dimensional ellipsoid. For the 2D ($n = 2$) case, for example, the shape of the contour will be an ellipse. In the context of this assignment, modeling 12 features would result in 12-dimensional Gaussian distribution.

The univariate Gaussian distribution is also uniquely defined by its standard deviation and its mean. In the multivariate case, it is defined by a mean vector μ and a covariance matrix Σ. Importantly, the distribution is also symmetric around its mean. Although the value of the distribution is technically non-zero everywhere, it will be practically zero when the distance of a value from the mean is more than a couple standard deviations. Another consequence of modeling using a Gaussian distribution is that outliers in the data will not be well represented.

2 In general, when do you expect that a Gaussian will work well and when do you think it will not work well?

A Gaussian classifier will work well when the data can be modeled with a Gaussian distribution, with the features listed in the previous question. This mainly means that the data is symmetric around its mean and has a Gaussian shape.

The main advantage that the Gaussian classifier potentially has is its speed at predicting categories for new observations. Indeed, unlike kNN, the Gaussian classifier does not have to search through the entire space of previously trained data to predict a category for new data. It simply must compute the unnormalized negative log likelihood (UNLL), given by $(x-\mu)\Sigma^{-1}(x-\mu)^T$, where x is the feature vector of the new data, μ is the mean vector and Σ is the covariance matrix describing the Gaussian distribution.

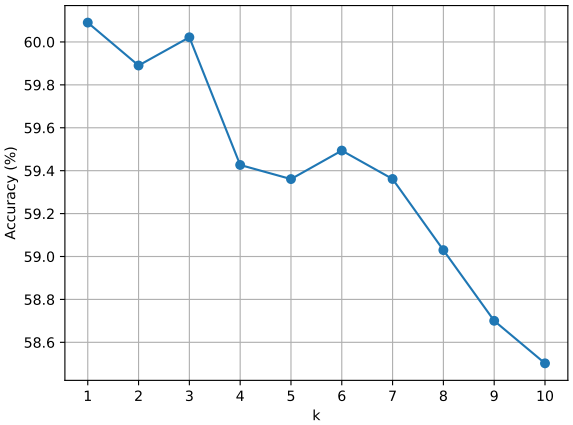

Figure 1: Accuracy of kNN (in percentage) versus k for local training data. The accuracy was average from performing k-fold cross-validation (with $k_{fold} = 10$).

The Gaussian classifier will not work in many situations. First, if the data itself cannot be modeled as Gaussian, then a Gaussian classifier will clearly not predict the correct values. For example, if the distribution is skewed in one direction or not symmetric, the classifier will not perform well.

Another example where a Gaussian model would not be appropriate is if there are multiple clusters of values. Here, multiple Gaussian distributions may be appropriate, but certainly not one.

Also, if the feature vector distributions of two categories are very similar, then their corresponding Gaussian distributions will be very close, perhaps intersecting. In this situation, a Gaussian classifier can have a hard time differentiating between these two categories.

A Gaussian classifier will also not work well for values that stray too far from the mean or with many outliers. In situations like this, kNN has the clear advantage, since it stores every training example, including outliers.

Examples of all the previously described situations will be given in Question 4.

Table 1: Confusion matrix for the Gaussian classifier.

	kids	classical	rnb	jazz	metal	pop	edm˙dance	latin	rock	country
kids	31	3	34	11	0	2	0	0	0	0
classical	16	54	0	7	0	0	0	0	0	0
rnb	3	0	72	2	0	0	0	0	0	0
jazz	2	0	11	66	0	0	0	0	0	0
metal	14	0	29	0	0	22	4	0	5	0
pop	1	0	68	0	0	1	0	0	0	0
edm˙dance	3	0	69	0	0	3	0	0	0	0
latin	0	0	70	0	0	0	0	7	0	0
rock	1	0	63	0	0	8	0	0	0	0
country	1	0	65	0	0	8	0	0	0	0

3 What values of k work best for the kNN classifier?

The results of testing kNN by k-fold cross-validation can be seen in Figure 1 and Table 2. The accuracy values were averaged from 10 different splits for each k, with k ranging from 1 to 10. It can be seen that the best value of k is 1, providing 60.090% accuracy.

Table 2: Accuracy of kNN versus k for local training.

k	Accuracy (%)
1	60.090
2	59.890
3	60.022
4	59.427
5	59.361
6	59.494
7	59.361
8	59.030
9	58.700
10	58.502

Similarly, the results of testing the kNN classifier on the Kaggle data set for k ranging from 1 to 3 can be seen in Table 3. Once again, $k = 1$ provides the best accuracy, now with 57.786%.

Table 3: Accuracy of kNN versus k on Kaggle.

k	Accuracy (%)
1	57.786
2	56.557
3	56.557

4 Based on your results from this assignment, which classifier (Gaussian, kNN, or other) works best for the task of music genre classification? Discuss why.

With 10-fold cross-validation testing on the provided test song data, an accuracy of 27.928% was achieved for the Gaussian classifier and 60.090% for the kNN classifier ($k = 1$). Similarly, on the Kaggle competition data set, an accuracy of 29.098% was achieved for the Gaussian classifier and 57.786% for the kNN classifier ($k = 1$). This implies that the kNN classifier is better for music genre classification. The reasons why will now be explored.

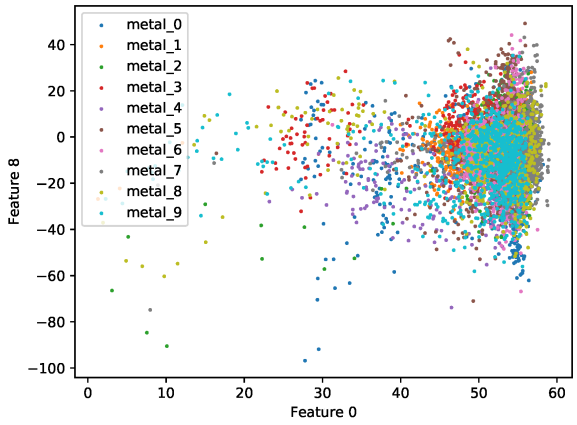

Figure 2: Feature 0 vs. feature 8 for 10 songs from the "metal" genre. The distribution is asymmetric.

To gain some insight into where both classifiers are succeeding and failing, a confusion matrix was created for both the Gaussian and kNN classifiers when performing local training. These can be seen in Tables 1 and 4, where each row in the table represents how many different genres were predicted (columns) for a given actual genre (row). We can see here that the kNN has relatively high numbers

Table 4: Confusion matrix for the kNN classifier, with $k = 1$.

	kids	classical	rnb	jazz	metal	pop	edm˙dance	latin	rock	country
kids	26	14	1	9	7	1	6	13	0	4
classical	0	76	0	1	0	0	0	0	0	0
rnb	0	2	29	8	2	7	7	21	0	1
jazz	0	2	1	73	2	0	1	0	0	0
metal	0	0	0	0	73	0	0	0	0	1
pop	0	0	9	7	9	5	14	17	1	8
edm˙dance	0	3	1	4	2	1	58	5	0	1
latin	0	0	0	0	0	0	4	72	0	1
rock	0	0	0	2	38	1	12	4	3	12
country	0	0	0	3	16	1	2	5	1	46

along the main diagonal of the matrix, which is a good sign. On the other hand, the Gaussian classifier mis-predicts many genres. It seems that almost all genres are predicted as "rnb", except for "classical" and "jazz", which are accurately predicted.

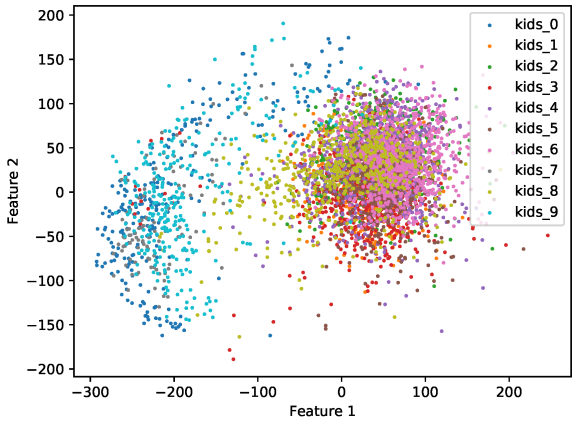

Figure 3: Projection of feature 1 vs. feature 2 for 10 songs from the "kids" genre. Two separate clusters of values can be seen, on the left and the right.

Most of the failures of the Gaussian classifier can be attributed to the simple fact that it retains much less information than the kNN for each genre. While the Gaussian classifier only stores a 12x12 covariance matrix and 12x1 mean vector for each genre, the kNN classifier stores every training example and uses these examples to predict every genre. This additional information can allow the kNN classifier to classify new songs with much greater accuracy.

Also, kNN can take into account very complex distributions of feature vectors, whereas the Gaussian classifier will attempt to fit a Gaussian distribution to the training data. This may not be appropriate if the data has any of the properties discussed in Question 2. Examples of these situations will now be provided, by examining relationships between two features in a song. Note that, although examining two features do not encompass the entire 12-dimensional distribution, it can still provide useful insight. Indeed, if the distribution of the 12 features is Gaussian, then necessarily any combination of two features will also be Gaussian.

Non-symmetric or skewed data is hard to model with a Gaussian distribution. This can be seen in Figure 2, where two features were plotted against each other for ten songs from the "metal" genre. It can clearly be seen that the data is skewed on one side, leading to asymmetry.

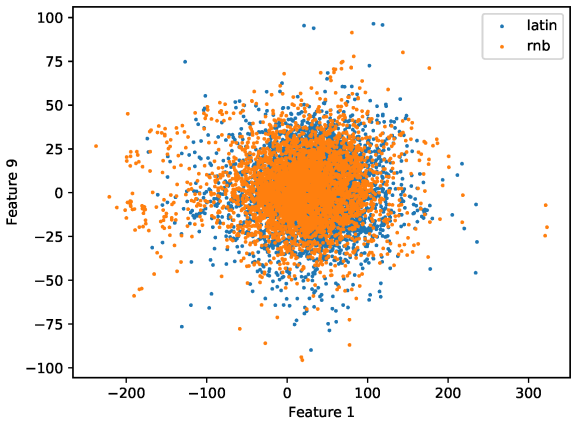

Figure 4: Projection of feature 1 vs. feature 9 for 5 songs from the "latin" genre and 5 songs from the "rnb" genre. The distributions are very similar.

Also, there are examples of genres where there are multiple clusters of data, which are hard to model with a single Gaussian. This can be seen in Figure 3, where two features were plotted against each other for ten songs from the "kids" genre. Two distinct groups of data can be seen.

There are also genres with very similar distributions which cannot be easily differentiated with

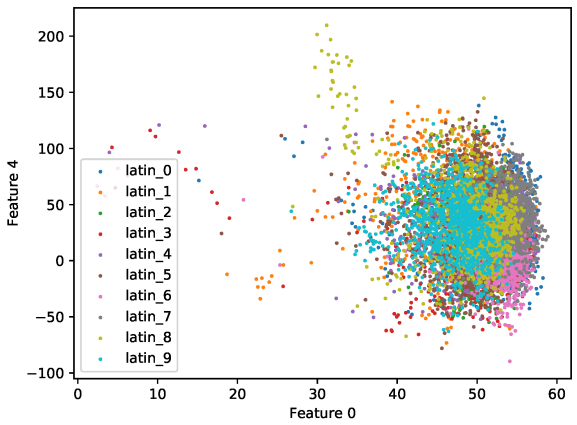

Figure 5: Projection of feature 1 vs. feature 2 for 10 songs from the "latin" genre. There are many extreme values for which kNN would have an easier time classifying genres than the Gaussian classifier would.

a Gaussian classifier. This can be seen in Figure 4, where the "latin" and "rnb" genres have very similar distributions. The similarity between the distributions for "rnb" and other genres is most probably the reason why so many genres were predicted to be "rnb" by the Gaussian classifier.

Also, distributions with many outliers cannot easily be classified with a Gaussian. This can be seen in Figure 5, where two features were plotted against each other for ten songs from the "latin" genre. These outliers can easily be accounted for with kNN.

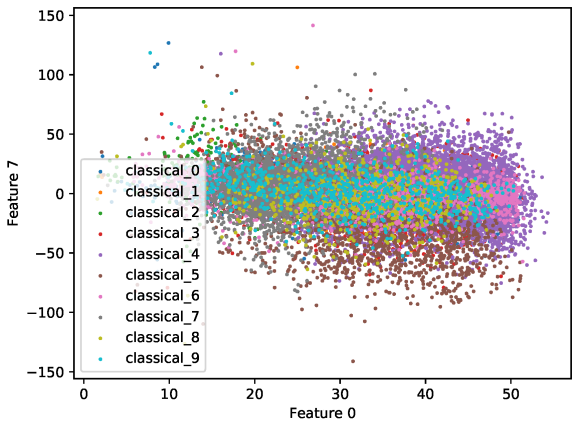

Figure 6: Projection of feature 0 vs. feature 7 for 10 songs from the "classical" genre. The distribution appears Gaussian.

Interestingly, the Gaussian classifier works relatively well for certain genres, such as "classical". This can be attributed to the fact that the "classical" data can be easily modeled with a Gaussian distribution. This can be seen for instance in Figure 6, which shows a plot of two features for ten songs from the "classical" genre.

The major downside to the kNN classifier for music classification is that it can be extremely slow. Using a k-d tree, like the one provided by the `scikit-learn` library, can greatly increase the speed of finding neighbours without sacrificing any accuracy.

Conclusion

The Gaussian and kNN classifiers were compared, showing the clear advantages of kNN for music genre classification. Possible further improvements to the work done in this assignment would be to implement a k-d tree from scratch or to refine the kNN algorithm, possibly using kernel functions to weight the neighbours.