COMP30027 Machine Learning: Project 1

THOMAS BLACK (1172648) & JOEL GRIFFIN (1072476)

Task 1. Pop vs. classical music classification:

Q1:

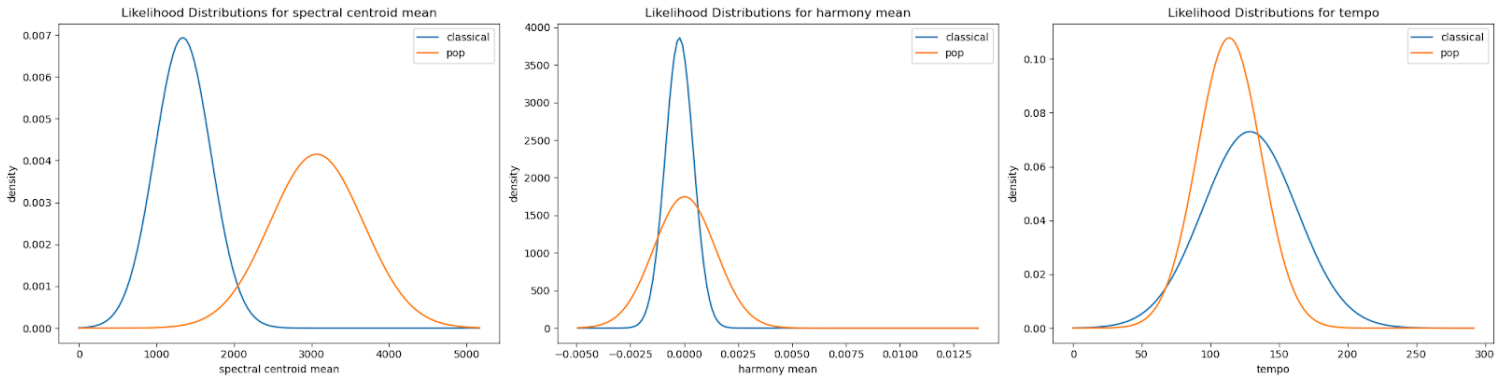
In this test our implementation of the Naive Bayes classifier was trained and tested against the “pop_vs_classical_train.csv” and “pop_vs_classical_test.csv” datasets respectively, with the positive class being “classical”. The performance of the model is summarised in the table below.

Performance Statistics of the Gaussian Naive Bayes Model on Task 1 Data

Metric	**Value**
Accuracy	0.9767
Precision	0.9524
Recall	1.0

The model seemed to perform well in this test, with the value of each metric being close to 1. Most notably, the recall is exactly 1 which indicates that the model correctly identified every instance of the positive class (classical).

Q2:



The probability density functions (PDFs) for the likelihoods of spectral centroid mean, harmony mean and tempo are shown in the above figure. If one attribute from the three was to be chosen to classify each music instance, the spectral centroid mean would be the best feature. Based on

the plots, the likelihood distributions for the spectral centroid mean are distinguishable from each other which makes the classification of pop or classical music easier to differentiate given the spectral centroid mean and reduces misclassification. In contrast, The likelihood distributions for harmony mean and tempo coincide with each other which makes it hard to distinguish pop and classical music and would cause a larger amount of errors in comparison to spectral centroid mean.

Task 2. 10-way music genre classification:

Performance Breakdown of the Gaussian Naive Bayes Model on Task 2 Data

	Recall	Precision	F1
Blues	0.158	0.429	0.231
Classical	0.850	0.895	0.872
Country	0.688	0.379	0.489
Disco	0.455	0.476	0.465
Hip-hop	0.286	0.500	0.364
Jazz	0.333	0.500	0.400
Metal	0.900	0.383	0.537
Pop	0.696	0.800	0.744
Reggae	0.642	0.5625	0.600
Rock	0.111	0.176	0.136
Overall (macro)	0.512	0.510	0.484

Total Accuracy: 0.495

Q3:

The 0-R baseline is implemented by determining the most frequent label in the training dataset and using that label to predict the music genre of the test data. If there were multiple most frequent labels, one of those labels would be randomly chosen. The average performance of this model (over 10 iterations) is summarised below.

Performance Breakdown of the 0-R Baseline

	Recall	Precision	F1
Reggae	1.0	0.070	0.131
Other Categories	0.0	-	-
Overall (macro)	0.100	0.070	0.131

Total Accuracy: 0.070

It should be noted that a precision and F1 score were not assigned to the other categories because it made no sense to calculate a precision score if all of the other categories were not predicted. From the performance results, reggae was the most frequent genre. In comparison to the original naive Bayes model, the 0-R model performs significantly worse. The performance of this model is expected since the model does not utilize any attributes of the music data.

The one-attribute baseline was implemented by considering only one feature at a time and using the associated Gaussian likelihood functions of the attribute and the prior probabilities to predict the test instances. Each attribute model was compared with each other using the total accuracy and whichever model had the highest accuracy would be the one-attribute baseline model. If there were ties between attributes, the first attribute that had the highest accuracy would be chosen. The performance of this model is summarised below.

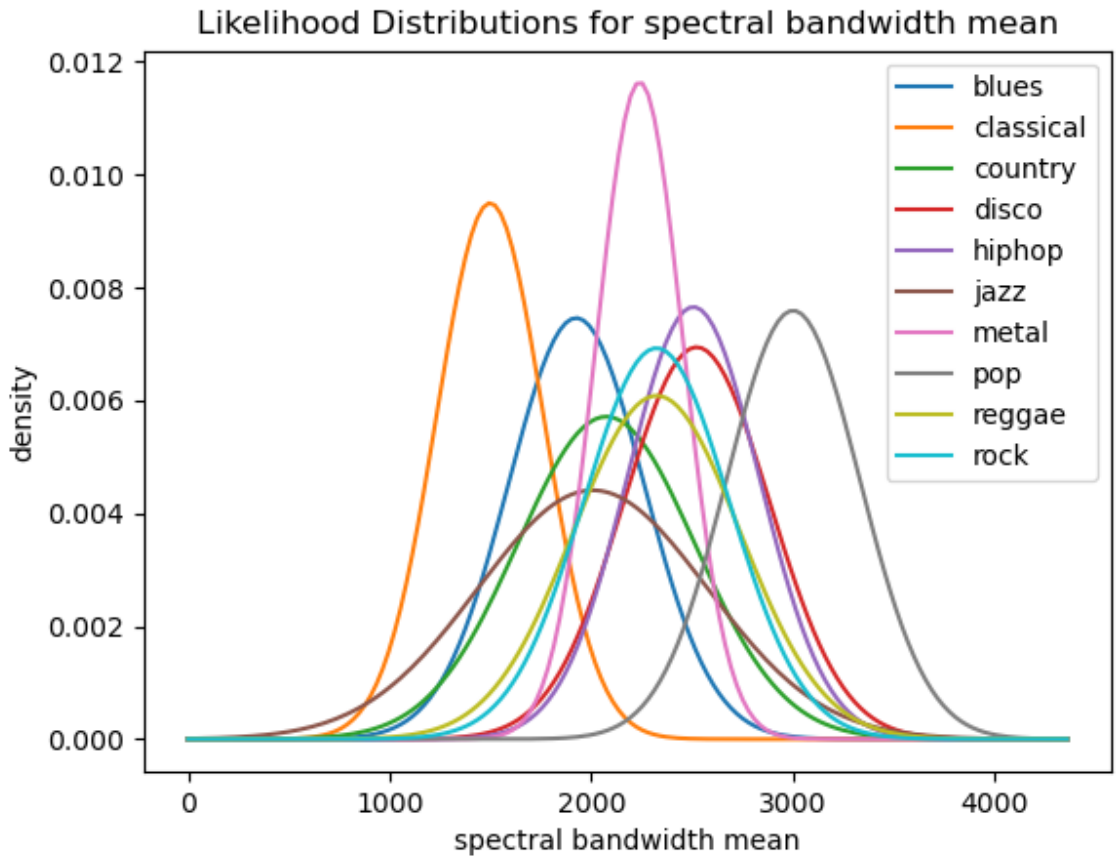
Performance Breakdown of the One-Attribute Baseline

	Recall	Precision	F1
Blues	0.211	0.174	0.190
Classical	0.750	0.469	0.577
Country	0.0	-	-
Disco	0.0	-	-
Hip-hop	0.381	0.276	0.320
Jazz	0.0	-	-
Metal	0.850	0.207	0.333
Pop	0.783	0.529	0.632
Reggae	0.0	-	-

Rock	0.0	-	-
Overall (macro)	0.297	0.331	0.410

Total Accuracy: 0.310

The one-attribute baseline chose to classify instances solely based on the spectral bandwidth mean. Some genres were not predicted due to their corresponding likelihoods being dominated by other likelihoods, which is evident in the following plot.



The original Gaussian naive Bayes model does perform better overall in comparison to the one-attribute baseline and can predict all music genres.

Q5:

For this question, a kernel density estimate (KDE) naive Bayes model was implemented using the Scikit-learn kernel density function. The kernel function used to generate the likelihood function was a Gaussian kernel and bandwidth for each KDE was derived using Silverman's rule

of thumb. The model was trained and tested on the ‘gztan_train.csv’ and ‘gztan_test.csv’ datasets respectively. The performance of this model is summarized below.

Performance Breakdown of the KDE Naive Bayes Model

	Recall	Precision	F1
Blues	0.263	0.714	0.385
Classical	0.900	0.947	0.923
Country	0.563	0.429	0.486
Disco	0.591	0.565	0.578
Hip-hop	0.429	0.529	0.474
Jazz	0.611	0.733	0.667
Metal	0.900	0.529	0.667
Pop	0.783	0.818	0.800
Reggae	0.643	0.409	0.500
Rock	0.333	0.450	0.383
Overall (macro)	0.602	0.612	0.586

Total Accuracy: 0.595

In comparison to the original Gaussian naive Bayes model, the KDE naive Bayes model performs better overall based on all the evaluation metrics. Furthermore, most categories had an increased recall and precision score, with the exception of the recall score for country and the precision score for reggae. The performance breakdown of this model implies that a Gaussian KDE is a more appropriate likelihood function for predicting music genres using naive Bayes. Despite the increased performance compared to other models, the model does misclassify 40% of the music instances, which suggests that a naive Bayes model may not be an appropriate machine learning algorithm for this dataset.

Silverman's rule of thumb formula was used to compute the bandwidth for each KDE function. The table below shows some summary statistics of the bandwidths.

Summary Statistics for the Bandwidths of the KDE Likelihoods

Mean	Standard Deviation	Min	25% Quantile	Median	75% Quantile	Max
7,303.34	46,683.53	0.000001	1.39	5.44	14.34	552,245.99

The statistics show that the bandwidth for each kernel function varies substantially. However, it makes a significant difference to alter the bandwidth for each feature/label pair. This is evident in the overall performance statistics for another KDE naive Bayes model that uses a constant bandwidth of 1 for each KDE likelihood. The overall performance of this model is significantly worse compared to the first KDE model.

Performance Statistics of KDE Model with Constant Bandwidth

	Recall	Precision	F1
Blues	0.053	0.050	0.051
Classical	0.400	0.320	0.356
Country	0.125	0.118	0.121
Disco	0.045	0.077	0.057
Hip-hop	0.095	0.100	0.098
Jazz	0.056	0.048	0.051
Metal	0.350	0.304	0.326
Pop	0.478	0.407	0.440
Reggae	0.214	0.230	0.222
Rock	0.074	0.095	0.083
Overall (macro)	0.189	0.175	0.181

Total Accuracy: 0.190

Q6:

For this question the Gaussian Naive Bayes Model was tested against the ‘gztan_test.csv’ dataset with various percentages (called drop rates) of missing values. Note that the model was trained without missing values. Missing values were handled by only calculating likelihoods for features which were present in a given instance. Consequently, drop rates of 0% and 100%

result in performances identical to that of Q3's full model and 0-R model, respectively (as seen in the experimental data below).

Overall Performance Statistics for Gaussian Model with Missing Test Attributes

Drop Rate		0%	10%	30%	50%	70%	90%	100%
Overall (Macro)	**Accuracy**	0.495	0.4890	0.4875	0.4790	0.4570	0.3595	0.0700
	Recall	0.5119	0.5033	0.5025	0.4901	0.4648	0.3624	0.1
	Precision	0.5101	0.5102	0.5049	0.4996	0.4675	0.3553	0.0700
	F_1	0.4838	0.4803	0.4773	0.4679	0.4385	0.3320	0.1308

In order to obtain meaningful results from tests with random elements, the performance of the model at a given drop rate was taken to be the average of its performances across 10 randomly altered data sets.

From drop rates of 0-70%, the value of each metric slowly decreased as drop rate increased. This was expected as having more missing values provides the model with less information, resulting in worse classifications. The slow rate of decrease could be explained by the presence of many uninformative features (features whose values were relatively evenly distributed across classes) whose loss would minimally affect the model's performance.

At drop rates 90% and 100% there were large decreases in performance across all metrics compared to previous drop rates. This could be explained by the exhaustion of 'buffer data' provided by uninformative features. At these high drop rates, the chance of a given instance having missing values in multiple informative features would be much higher than at low drop rates, resulting in substantially worse classifications.

Full Performance Statistics for Gaussian Model with Missing Test Attributes

Drop Rate		0%	10%	30%	50%	70%	90%	100%
Blues	**Recall**	0.1579	0.1947	0.2053	0.1947	0.2	0.1316	-
	Precision	0.4286	0.4734	0.4474	0.4617	0.3928	0.2386	-
	F_1	0.2308	0.2554	0.2791	0.2703	0.2619	0.168	-
Classical	**Recall**	0.8500	0.8350	0.84	0.815	0.78	0.585	-
	Precision	0.8947	0.9095	0.8955	0.8693	0.8007	0.5631	-
	F_1	0.8718	0.8700	0.8655	0.8401	0.7889	0.5715	-

Country	**Recall**	0.6875	0.6687	0.6188	0.6	0.5687	0.3937	-
	Precision	0.3793	0.3725	0.359	0.3227	0.2782	0.1959	-
	F_1	0.4889	0.4779	0.4535	0.4192	0.3731	0.2587	-
Disco	**Recall**	0.4545	0.4000	0.4	0.3636	0.3136	0.2318	-
	Precision	0.4762	0.4477	0.439	0.4184	0.3956	0.3052	-
	F_1	0.4651	0.4218	0.4179	0.3866	0.3493	0.262	-
Hiphop	**Recall**	0.2857	0.2714	0.2524	0.2762	0.2143	0.1429	-
	Precision	0.5000	0.4433	0.4822	0.528	0.46	0.3847	-
	F_1	0.3636	0.3364	0.3305	0.3602	0.2907	0.2069	-
Jazz	**Recall**	0.3333	0.3722	0.35	0.3389	0.2944	0.2111	-
	Precision	0.5	0.5497	0.5473	0.5133	0.505	0.3586	-
	F_1	0.4000	0.4433	0.4252	0.4063	0.3686	0.2633	-
Metal	**Recall**	0.9000	0.9000	0.9	0.895	0.89	0.85	-
	Precision	0.3830	0.3959	0.3933	0.4036	0.4172	0.3616	-
	F_1	0.5373	0.5498	0.547	0.5562	0.5677	0.5066	-
Pop	**Recall**	0.6957	0.7130	0.7217	0.7261	0.7261	0.6261	-
	Precision	0.8000	0.7922	0.7987	0.7554	0.7315	0.5858	-
	F_1	0.7442	0.7503	0.7578	0.7394	0.7277	0.6028	-
Reggae	**Recall**	0.6249	0.5643	0.6143	0.5357	0.4714	0.3143	1.0
	Precision	0.5625	0.5366	0.4985	0.5063	0.4162	0.3015	0.0700
	F_1	0.6	0.5474	0.5486	0.5191	0.4334	0.3027	0.1308
Rock	**Recall**	0.1111	0.1296	0.1222	0.1556	0.1889	0.137	-
	Precision	0.1765	0.1809	0.1881	0.2175	0.2781	0.2576	-
	F_1	0.1364	0.1507	0.1474	0.1812	0.2239	0.1771	-

References:

- *sklearn.neighbors.KernelDensity*. (n.d.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KernelDensity.html#sklearn.neighbors.KernelDensity
- Wikipedia contributors. (2023, January 29). *Kernel density estimation*. Wikipedia. https://en.wikipedia.org/wiki/Kernel_density_estimation