PART 1

Exploring Data

In our data exploration process, We have to investigate each dataset individually to highlight the unique patterns between our features. But furthermore, we also have to assess the dataset as a whole by viewing patterns on a concatenated dataset, as these 10 datasets only address their respective genres, and the correlated dataset will better indicate an overall trend or pattern between the features, and their relative strength to help define the genres.

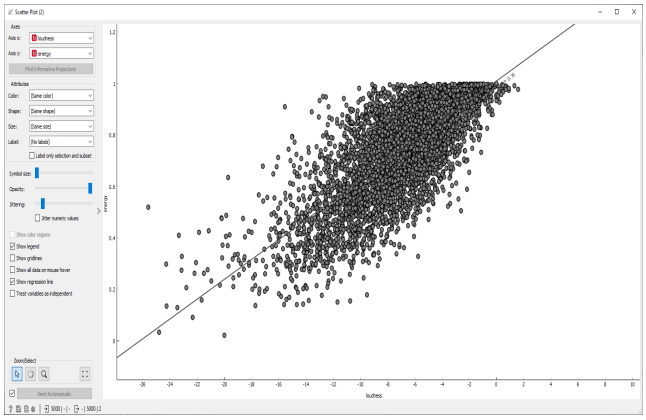
Initial preprocess exploration

>Each dataset contains 5000 instances of songs from their respective genre. From the features there are 3 categorical (key, mode, obtained date) and 11 numerical values (popularity, acousticness, danceability, duration_ms, energy, instrumentalness, liveness, loudness, speechiness, tempo, valence) and from the meta's there are 1 categorical (music genre), 1 numeric (instance_id) and 3 strings (artist_name, track_hash, track_name).
>The combined data set will contain 50000 instances - this is large enough to ensure we do not require cross validation.
>The scale of the features are different and we will likely require normalisation.
>Our target class genre is equally distributed, there is 10% of data missing from the feature 'Tempo'
>Popularity appears to be the most important feature whilst liveliness tends to be the least.

For Orange, we have to specify that the "music_genre" column is the target class, as initially, Orange will miss this because each dataset only has 1 class for "music_genre" for each file, corresponding to the music genre of the file.

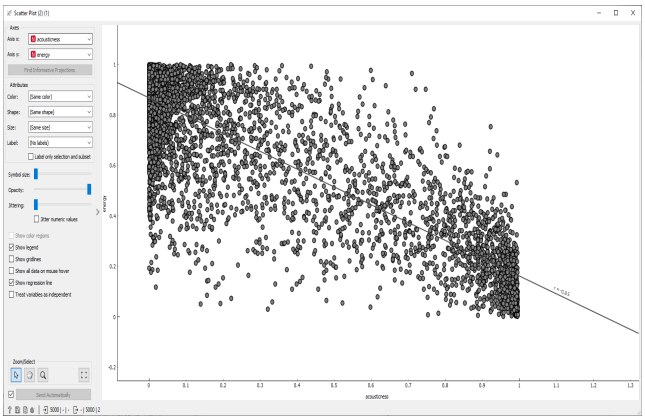
Important patterns:

(1) Correlation:



Energy and Loudness has a strong positive correlation (over 0.75). This makes sense because songs with high energy tend to be louder in volume (i.e higher presence of Bass, stronger Vocals). The Scatter plot for Electronic.csv below shows a correlation coefficient of r = 0.76

(2) Correlation:



On the other hand, Acousticness has a strong inverse relation (under -0.75) with Energy and Loudness. Accoustiness's inverse relationship also makes sense, as tracks dominant with acoustic instruments (instead of digital instruments such as Synthesizers and Electric Guitars) tend to be softer and are more relaxing to listen to. The Scatter plot for Anime.csv below shows a correlation coefficient of r = -0.85

We can see these sets of correlations prevail in the unique datasets as well as the concatenated dataset, scoring high on the correlations (generally over 75%) for the relationships mention above

Runner-ups for some correlation features are:

>Danceability - Valence : Valence is a measure of how “happy” a song feels, and it's in human nature to feel more inclined to dancing when we are happy.

- above 50% correlation are: Anime, Classical,

>Instrumentalness - Inverse to Loudness: Instrumentalness measures how much vocals are present in a track, therefore it makes sense that sounds with strong vocal presence (has a low instrumentalness score) will be louder as the singers voice will be the loudest part of a song.

- Strongest correlations (above 40%) are: Anime, Electronic Music.

We see these runner-up correlations prevailing their strength overall, as we can see that they have a 40-55% correlation in the concatenated dataset.

Impact:

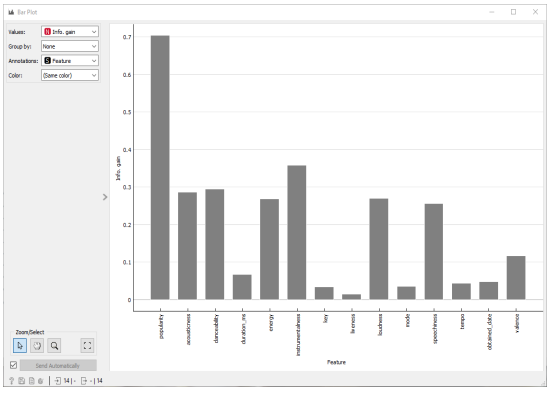
The dependencies among these features as analysed above can negatively impact our learning model due to “multicollinearity”, which can cause misleading and skewed results on a predictive model as it causes a model’s coefficients and weights to be very sensitive (overfitting), which will reduce predictive performance when ran on a different dataset (our test dataset).

We will have to address this and either:

>Create a PCA to reduce the number of features and have the Principle Components be more independent of each other, creating better variances for our predictive model.

>Remove the strongly correlated variables. As with strongly correlated variables, we can infer one dependent variable from another, which means not all the variables are needed.

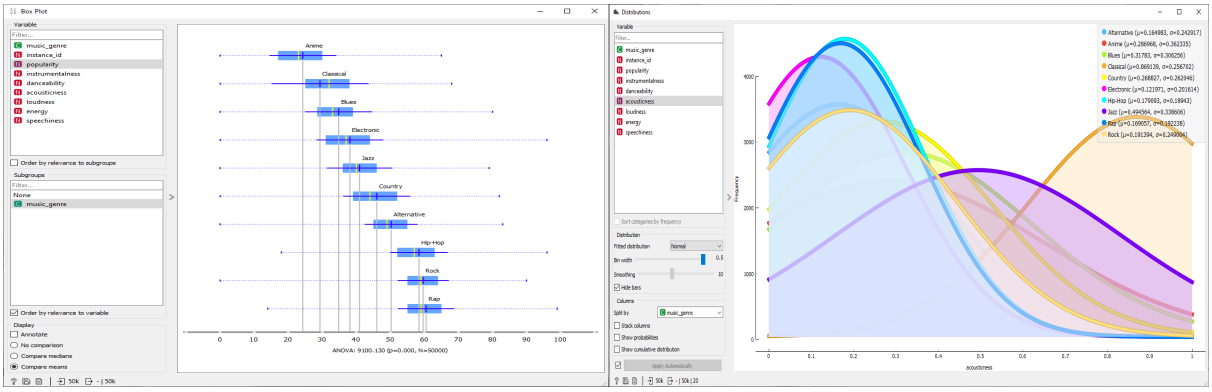
(3) Feature ranking:



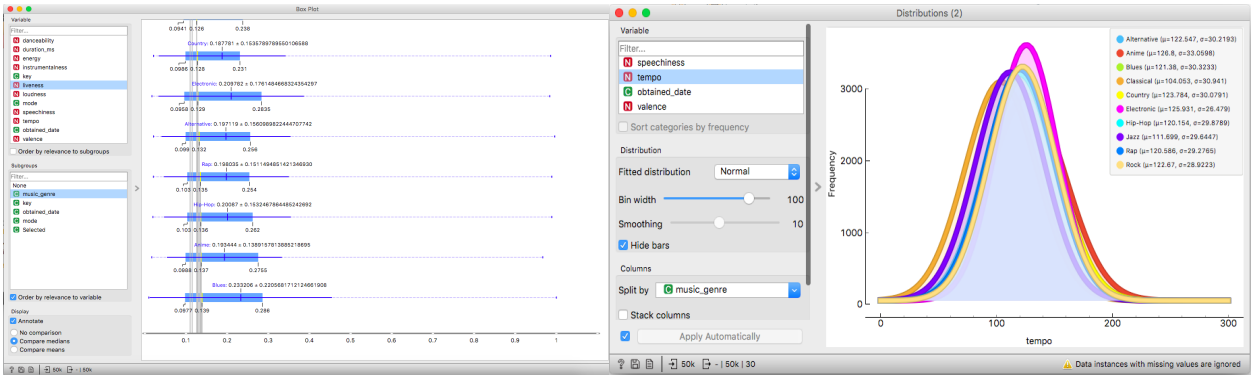
We cannot rank the audio features within each dataset as there is only 1 class per dataset. To be able to rank, we have to concatenate the datasets into 1 table with multiple classes, as that will allow us to see which features are the best at defining a genre, and which features are the worst for us to filter out.

We were able to rank & display the features that contributed most to the target class (genre) thanks to the Rank and Bar Plot widgets, and can see that “Popularity” stands out by a far margin to the other features in terms of information gain, followed by Instrumentalness, Dancablity & Accousticness.

To further cement this result, Box Plots and Distribution graphs show a fine distinction between the genres when we define the values as Popularity (left) and Accousticness (right) as a visual example.



However, lower ranked features tend to demonstrate less of a distinction between genres for example liveness(left) and tempo(right).



Impact:

The features which contribute a low amount of information (provides little distinction between genres) create unnecessary noise (confusion). Removing non-informative features can reduce noise, and can increase the contrast between genres.

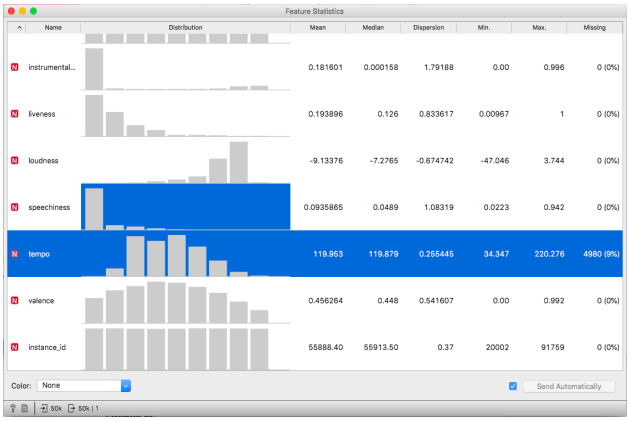
We will have to address this and either:

>Create a PCA to reduce the number of insignificant features.

>Determine relevant features using tree and remove insignificant features.

>Remove the variables which make insignificant contributions to info gain.

(4) Missing data:



All the missing data can be found in the feature tempo. All other information in all other features are complete.

Impact:

Running the model with missing values will affect the accuracy of our model.

We will have to address this and either:

>Remove tempo.

>Remove or impute missing values.

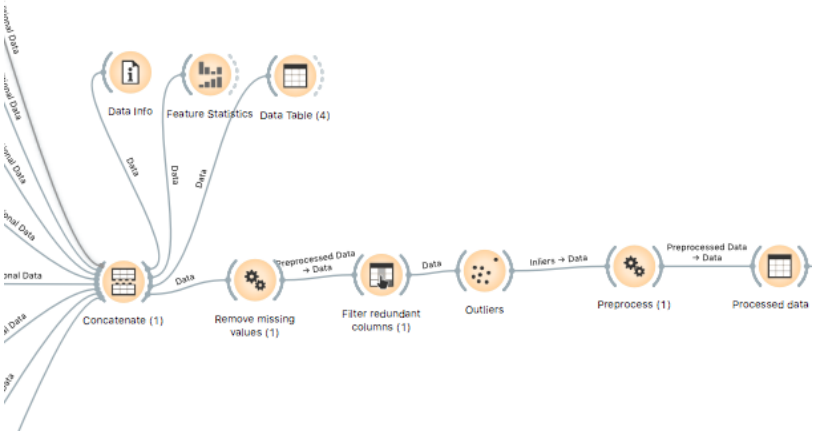
PART 2

Initial design

>For our initial design we knew that we needed to combine all the genres into one dataset to allow our algorithm to learn from all types of genre by making distinctions between the features of these instances as determined by our initial ranking of our features and exploration of correlation between features as determined by our initial exploration of correlations. To do this we use the 'Concatenate' tool.

>From the initial data exploration the "Data info" tool indicated that we had instances with missing values and the 'feature statistic' tools indicate that 10% of data is missing in tempo. To remedy this, we remove instances with missing values using the 'Preprocess' tool in orange as replacing these missing values could skew the accuracy of our predictions if we trained our model using randomised values.

>To make our dataset easier to work with we removed redundant columns (artist_name, track_hash, track_name) using the 'Select Column' tool.
>For each genre when we compared their features in the initial data exploration, the shape of the box plots indicate the potential of outliers - this is because the tail on either side of the box (each representing 25% of the data on either sides of the middle 50%) tend to be quite long which may mean that there are some instances of outliers in the dataset. To remedy this we use the 'Outlier' tool in orange to remove outliers in the dataset.



Modification
For our initial test we used the method above to obtain the processed data. We then experimented with various machine learning techniques (KNN, Random Forest, SVM, Logistic Regression, Naive Bayes, Neural Network) to determine the best Machine learning algorithm to accurately predict our data using a 'test and score' tool to rank the techniques based on its accuracy.
Our top 3 most accurate machine learning tools were neural networks (46.4%), logistic regression (45%) and naives bayes (42%).

Removing features
We then experimented with removing features to increase the accuracy score of these machine learning techniques.
From our initial exploration we found that the features Energy, Loudness and Acousticness had a high correlation. To better our machine learning models we should remove one if not more of these features as removing non-informative features can reduce noise, and can increase the contrast between instances.

To determine the features we needed to remove we used the 'Tree' tool and info gain rank ('Rank' tool). On the 4th level of the tree we found Popularity, acousticness, danceability, instrumentalness, loudness, speechiness, valence.
This suggested we could reduce the number of our features by removing Duration_ms, energy, liveness and tempo. This is supported by the info gain rank of these features as these features had lowest info gain of all the features.

After removing these features we observed an increase of our accuracy score on kaggle. This is because we were able to minimise noise and concentrate on features which hold distinction between genres.
This also meant we could rearrange our preprocessing method. As all missing values were caused by tempo - we instead removed tempo after concatenating which meant we did not need to remove any instances allowing all genres to have an equal amount of instances.

Experimenting with different values for Neural Networks (NN)
After reducing the number of features, it was clear to us that NN would perform most accurately when predicting data. To enhance its performance we investigated different values for Neurons, Activations, Solvers, Regularization and Maximum number of iterations.

We found that this was the best combination

Hidden layers perform nonlinear transformations of the inputs entered into the network.
Neurons: 50 (tested values between 50-1000) - trial and error (online resources recommend experimenting (lower numbers tend to be better).
https://www.researchgate.net/post/How-to-decide-the-number-of-hidden-layers-and-nodes-in-a-hidden-layer

The choice of activation function in the hidden layer will control how well the network model learns the training dataset.
Activations: reLu (tested ReLu, Tanh, Log, identity) - trial and error (online resources recommend experimenting (ReLu however overcomes disappearing gradient issue)).
https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/

A stochastic gradient descent based optimizer for optimizing the parameters in the computation graph.
Solvers: Adams (tested Adams, SgL, L-BFGS-B) - trial and error in accordance with online resources.
https://www.solver.com/training-artificial-neural-network-intro

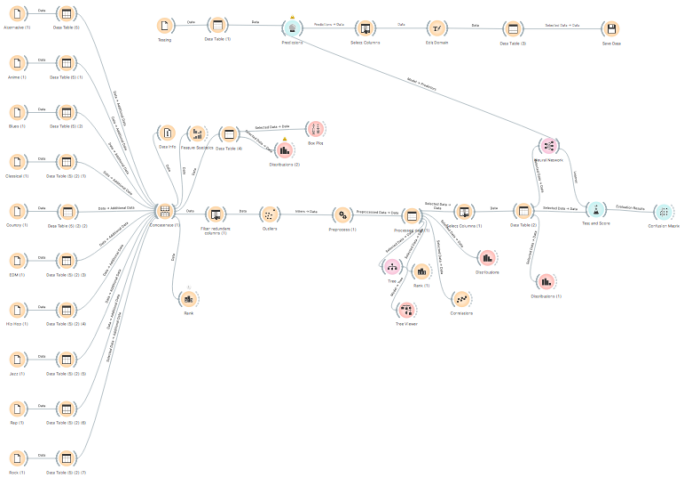
Regularization is a technique used for tuning the function by adding an additional penalty term in the error function (prevents overfitting and improves generalisation)
Regularization: 0.0001 (tested 0.0001-20) - trial and error in accordance with online resources.
https://www.analyticsvidhya.com/blog/2018/04/fundamentals-deep-learning-regularization-techniques/

Describes the number of times a batch of data passed through the algorithm.
Maximum number of iterations: 300 (tested 100-500) - trial and error in accordance with online resources.
https://www.researchgate.net/post/What_is_the_optimal_number_of_iterations_in_a_neural_network_which_also_avoids_over-fitting

Experimenting with different values for Outliers
After maximising the performance of NN we tried to enhance our model by adjusting preprocessing values this was limited to outliers.

Contamination is the proportion of outliers in the dataset
There are no resources on the best value for the contamination percentage. By exploring values between 10-50% we found that 20% performed the best with our data. The parameter Local Outlier Factor was used, this method obtains local density from the k-nearest neighbors of 20.

Final model



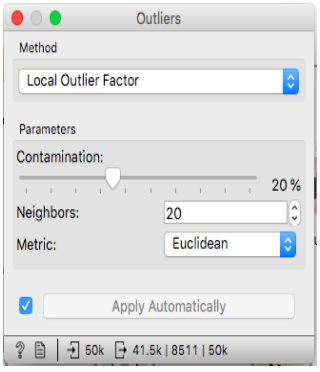
With our final model we were able to obtain the result of 56.514% on kaggle.
We chose this system as we felt that this model would most accurately categorise our test instances into the correct genre and demonstrated an optimised neural network algorithm using the provided data set.

	Predicted										
	Alternative	Anime	Blues	Classical	Country	Electronic	Hip-Hop	Jazz	Rap	Rock	Σ
ative	44.9 %	1.0 %	1.6 %	0.3 %	13.2 %	5.2 %	8.5 %	6.5 %	5.8 %	15.1 %	108807
nime	3.5 %	72.5 %	8.9 %	7.1 %	4.6 %	5.1 %	0.0 %	1.9 %	0.1 %	0.4 %	93211
lues	5.7 %	11.8 %	58.4 %	1.7 %	11.2 %	5.7 %	0.3 %	12.2 %	0.3 %	3.5 %	104717
sical	2.9 %	4.5 %	2.8 %	82.2 %	0.7 %	2.3 %	0.0 %	5.0 %	0.0 %	0.3 %	96052
ntry	11.6 %	1.8 %	6.6 %	0.1 %	51.8 %	2.7 %	1.7 %	7.5 %	1.4 %	13.9 %	109337
ronic	9.0 %	6.4 %	7.5 %	0.9 %	4.2 %	61.8 %	2.1 %	12.1 %	1.9 %	2.1 %	104383
-Hop	4.3 %	0.0 %	0.1 %	0.0 %	0.7 %	1.4 %	45.8 %	1.1 %	39.7 %	4.1 %	107276
Jazz	5.8 %	1.7 %	13.8 %	7.6 %	7.9 %	14.7 %	2.1 %	50.6 %	0.6 %	3.0 %	105010
Rap	3.2 %	0.0 %	0.1 %	0.0 %	0.4 %	0.6 %	36.5 %	0.6 %	43.8 %	8.6 %	105091
Rock	9.0 %	0.3 %	0.3 %	0.2 %	5.2 %	0.6 %	2.9 %	2.6 %	6.5 %	49.2 %	103366
Σ	92499	86550	89577	95892	109469	95744	115692	95379	102810	153638	1037250

Observing the confusion matrix, we can see that genres with lower accuracy tend to be caused by 'ambiguous' genres - for example distinctions between hiphop and rap songs can be minute (often indistinguishable in certain circumstances even to humans)- and hence in these genre we see a lower performance in accuracy (e.g 43.8% prediction accuracy in rap) however in unique genres such which is more difficult to misclassify we see a higher performance in accuracy (e.g 82.2% accuracy in classical).
After modifying the values within NN and outliers - we felt that these results gave us the most accurate model to predict our test data.

<u>Interesting features of the model</u>

Outlier



Interestingly we found that increasing the percentage of outliers from 10% to 20% improved the accuracy of our test however - any % above or below 20% decreased our accuracy.

It is likely that any value below 20% introduces outliers creating noise and inaccuracy in the model, however any values above 20% removed valuable instances.

Features

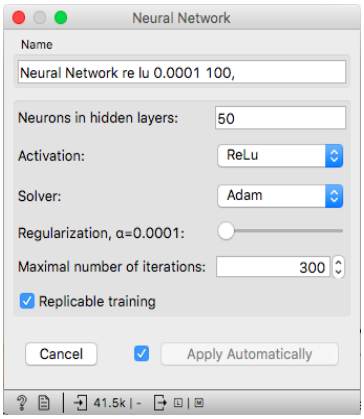
Data Table (2)

	music_genre	instance_id	popularity	instrumentalness	speechiness	loudness	acousticness	danceability	valence
1	Alternative	34428	0.4948	0.0843373	0.0104495	0.810634	0.360442	0.392271	0.278788
2	Alternative	76036	0.4742	2.62048e-06	0.100903	0.813879	0.300201	0.7269	0.375758
3	Alternative	37341	0.4433	0	0.0165451	0.839637	0.298193	0.654577	0.626263

We began with 11 features and were able to reduce this to 7.

Introducing Duration_ms, energy, liveness and tempo reduces or removing any of the above 7 features the accuracy of our model. From this we know that we've removed unnecessary noise from 'low information' variables and reduced multicollinearity in our final model. As a result we have only included features which help our model distinguish an instance's genre.
These are:
Popularity, Instrumentalness, Speechiness, Loudness, Acousticnesss, Dancibility, Valence

Machine learning algorithm



We found that Neural Networks produced the most accurate results for our test data. Neural network process: once an input layer is determined, weights are assigned. All inputs are then multiplied by their respective weights and then summed. Afterward, the output is passed through an activation function, which determines the output. If that output exceeds a given threshold, it "fires" (or activates) the node, passing data to the next layer in the network.

Decreasing Neurons in the hidden layer from 100 to 50 increased the accuracy of our model. This suggests values over 50 overfit to the training set and thus unable to efficiently generalize to new unseen data. Values below 50 under fits the data as the model does not categorise genres as well as it could.

Activations, solver and regularization remain at their default setting - experimenting with other values decreased the accuracy of our models which is likely caused by the disappearing gradient issue which ReLu overcomes.

The Number of iterations was increased from 200 to 300 which was what produced the most accurate result for our model. This suggests values over 300 overfit to the training set and thus unable to efficiently generalize to new unseen data. Values below 50 under fits the data as the model does not categorise genres as well as it could.

<u>Challenge</u>
Ease of interpretation
Observing the confusion matrix:

	Predicted										
	Alternative	Anime	Blues	Classical	Country	Electronic	Hip-Hop	Jazz	Rap	Rock	Σ
ative	44.9 %	1.0 %	1.6 %	0.3 %	13.2 %	5.2 %	8.5 %	6.5 %	5.8 %	15.1 %	108807
nime	3.5 %	72.5 %	8.9 %	7.1 %	4.6 %	5.1 %	0.0 %	1.9 %	0.1 %	0.4 %	93211
Blues	5.7 %	11.8 %	58.4 %	1.7 %	11.2 %	5.7 %	0.3 %	12.2 %	0.3 %	3.5 %	104717
sical	2.9 %	4.5 %	2.8 %	82.2 %	0.7 %	2.3 %	0.0 %	5.0 %	0.0 %	0.3 %	96052
untry	11.6 %	1.8 %	6.6 %	0.1 %	51.8 %	2.7 %	1.7 %	7.5 %	1.4 %	13.9 %	109337
ronic	9.0 %	6.4 %	7.5 %	0.9 %	4.2 %	61.8 %	2.1 %	12.1 %	1.9 %	2.1 %	104383
-Hop	4.3 %	0.0 %	0.1 %	0.0 %	0.7 %	1.4 %	45.8 %	1.1 %	39.7 %	4.1 %	107276
Jazz	5.8 %	1.7 %	13.8 %	7.6 %	7.9 %	14.7 %	2.1 %	50.6 %	0.6 %	3.0 %	105010
Rap	3.2 %	0.0 %	0.1 %	0.0 %	0.4 %	0.6 %	36.5 %	0.6 %	43.8 %	8.6 %	105091
Rock	9.0 %	0.3 %	0.3 %	0.2 %	5.2 %	0.6 %	2.9 %	2.6 %	6.5 %	49.2 %	103366
Σ	92499	86550	89577	95892	109469	95744	115692	95379	102810	153638	1037250

we can see that genres with lower accuracy tend to be caused by 'ambiguous' genres - for example distinctions between hiphop and rap songs can be minute (often indistinguishable in certain circumstances even to humans) - and hence in these genre we see a lower performance in accuracy (e.g 43.8% prediction accuracy in rap) however in unique genres such which is more difficult to misclassify we see a higher performance in accuracy (e.g 82.2% accuracy in classical).

As a result it is relatively easy to interpret the predictions made by the model. Where the distinction between genres is apparent - our model makes predictions reflective to this (for example in classical no rap songs and only 0.2% of hip hop songs are misclassified as classical) however where the distinction between genres is harder to determine our model's accuracy decreases (e.g 43.8% prediction accuracy in rap with 38.7% of rap songs being misclassified as hiphop).

Limitation

Our current model is able to accurately predict genres of instances that have a clear relationship to a certain genre however it is weak at predicting instances which are more ambiguous.
Because there are no apparent features which assist us in distinguishing ambiguous genres, it is quite difficult to make our model more accurate as we are limited by the nature of the dataset and the orange application.

For example:
If we were to introduce more features - we would create noise and reduce the accuracy of the model.
If we were to remove more features - we would not have enough information to distinguish between other genres.

Comparison

Neural Network (NN) process:
Once an input layer is determined, weights are assigned. All inputs are then multiplied by their respective weights and then summed. Afterward, the output is passed through an activation function, which determines the output. If that output exceeds a given threshold, it "fires" (or activates) the node, passing data to the next layer in the network.

K-nearest Neighbors (KNN) process:
KNN plots all the instances on a graph and based on the assumption 'similar things exist in close proximity' assigns predictions based on clusters of instances.

Random Forest (RF) process:
Random Forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

Simple Model (KNN)
NN performs more accurately than KNN. NN uses weights of inputs to mathematically identify relationships between features and associate them to a genre whereas KNN assumes a relationship between instances based on proximity of plotted instances. Therefore when comparing KNN to NN, we are less confident with the predictions of KNN as the weight of the various features of the inputs are disregarded and the classification of instances of genre are made based on an instance proximity rather than its feature's relationship to a genre (harder to explain why an instance is classified into a genre)

Complex model (RF)
NN performs more accurately than RF, Both algorithms have the ability to produce relatively accurate predictions however in this dataset NN tends to create more accurate predictions. This may be attributed to the number of features involved in classifying an instance into a genre and the number of possible prediction outcomes. Therefore when comparing RF to NN, we are less confident with the predictions of RF as the size of the features make these trees extremely complex and the number of possible predictions make the outcome less accurate when the final prediction is made.

Summary

Overall, I would trust the predictions of the final model (Neural Network). Whilst there is a higher likelihood of misclassification for certain genres, these misclassified instances tend to be of genres which share similar features. This means misclassified instances tend to be acceptable recommendations (e.g a user listening to rap songs will also likely enjoy hip hop songs).