Exploratory Data Analysis on EDM

Tega Ewubare

(08 June, 2024)

## Packages

# List of required packages  
  
library(tidyverse)  
library(knitr)  
library(paletteer)  
library(factoextra)  
library(ggpubr)  
library(PerformanceAnalytics)  
library(tools)

## Data Import

edm\_data = read\_csv("train\_data\_final.csv")

## Introduction

Electronic Dance Music (EDM) is a compilation of electronic music subgenres that have developed over the last few decades EDM has become a global phenomenon, with festivals and concerts drawing millions of fans each year. Here in Miami, the Ultra Music Festival is an annual event that has become a staple of the city’s music scene and a favourite of many students. Given the genre’s popularity, it is essential to understand the distinguishing characteristics of each subgenre to more quickly identify the subgenre and better appreciate the diversity of EDM.

## Dataset

This dataset comes from Sivadithiyan official on Kaggle. To collect this dataset, the author used YouTube music mixes and Ableton to extract 2000 audio clips per subgenre.

## Goal

This project aims to explore the audio features of EDM genres using a dataset containing 16 different subgenres to determine which feature best correlates with the genre to predict genres.

**Research Question**

What are the distinguishing audio features of each EDM genre in the dataset?

There are 131 columns in the dataset. We can use all of them except ‘label’ for our analysis. The ‘label’ column contains the names of subgenres. I will be using the following columns for the analysis: rmse\_mean, spectral\_centroid\_mean, spectral\_bandwidth\_mean, rolloff\_mean, zero\_crossing\_rate\_mean, chroma\_cqt\_mean, and spectral\_contrast\_mean. These columns contain audio features that are the easiest to interpret and understand.

**Null Hypothesis**

The zero crossing mean rate is not significantly different across the EDM genres in the dataset.

**Alternative Hypothesis**

The zero crossing mean rate is significantly different across the EDM genres in the dataset.

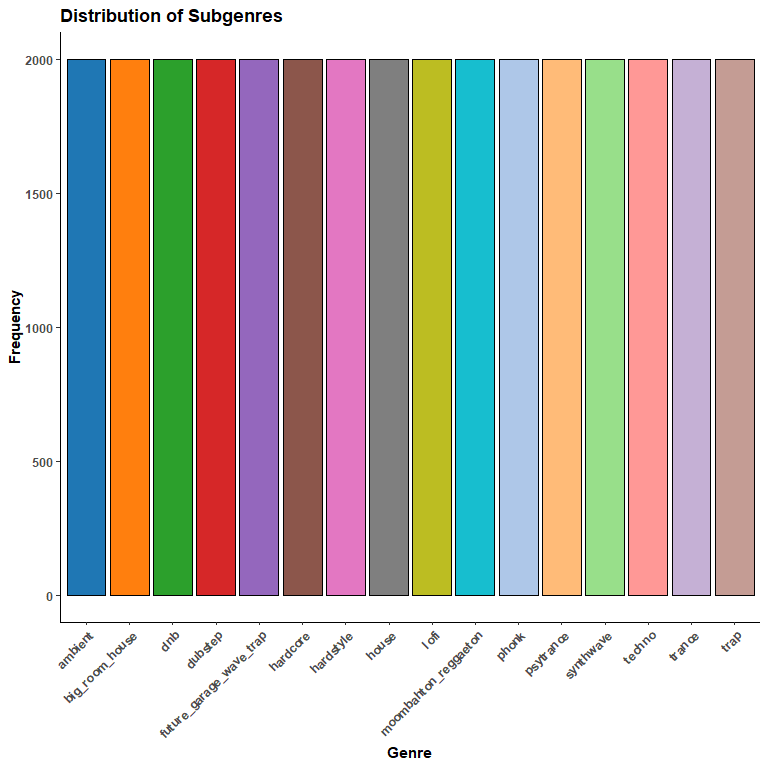
## Variables and Description

There are 131 columns in the dataset. We can use all of them except ‘label’ for our analysis. The ‘label’ column contains the names ofsubgenre. I will be using the following columns for the analysis: rmse\_mean, spectral\_centroid\_mean, spectral\_bandwidth\_mean, rolloff\_mean, zero\_crossing\_rate\_mean, chroma\_cqt\_mean, spectral\_contrast\_mean. These columns contain audio features that are the most easy to interpret and understand.

* rmse\_mean: Average root mean square error, reflecting the average energy of the audio signal.
* rmse\_std: Standard deviation of the root mean square error, indicating variability in signal energy.
* spectral\_centroid\_mean: Mean value of the spectral centroid, associated with perceived brightness of sound.
* spectral\_centroid\_std: Standard deviation of the spectral centroid, showing brightness variability.
* spectral\_bandwidth\_mean: Average spectral bandwidth, measuring spread around spectral centroid affecting timbre.
* spectral\_bandwidth\_std: Standard deviation of spectral bandwidth, indicating variability in frequency spread.
* rolloff\_mean: Mean spectral rolloff point, frequency below which a specified percentage of spectral energy is contained.
* rolloff\_std: Standard deviation of the spectral rolloff, showing variability in frequency distribution cutoff.
* zero\_crossing\_rate\_mean: Average rate at which signal changes from positive to negative or vice versa.
* zero\_crossing\_rate\_std: The zero crossing rate’s standard deviation indicates variability in textural features.
* mfcc1\_mean to mfcc40\_mean: Each mean of Mel-frequency cepstral coefficients, capturing aspects of sound’s shape and texture.
* mfcc1\_std to mfcc40\_std: Each standard deviation of Mel-frequency cepstral coefficients, describing variability in each coefficient.
* chroma1\_mean to chroma12\_mean: Mean values for the twelve different chroma features relating to the twelve pitch classes.
* chroma1\_std to chroma12\_std: Standard deviations for the twelve chroma features, indicating variability in harmonic content.
* tonnetz1\_mean to tonnetz6\_mean: Mean values of tonal centroid features, capturing harmonic and tonal characteristics.
* tonnetz1\_std to tonnetz6\_std: The tonal centroid features’ standard deviations show harmonic variability.
* chroma\_cqt\_mean: Mean of constant-Q chroma features, robust representation of harmonic content over time.
* chroma\_cqt\_std: The standard deviation of constant-Q chroma features indicates variability in harmonic patterns.
* spectral\_contrast\_mean: Average spectral contrast, measuring dynamic range between spectral peaks and valleys.
* spectral\_contrast\_std: Standard deviation of spectral contrast, representing variability in dynamic range across clips.
* label: Categorical variable containing genre labels for classifying music into different genres.

First, let us show the dataset’s subgenres and the number of audio clips for each genre.

# The syntax below will create a bar plot of the distribution of subgenres in the dataset  
  
ggplot(edm\_data, aes(x = label, fill = label)) +   
 geom\_bar(color = "black") +   
 scale\_fill\_paletteer\_d(palette = "ggsci::category20\_d3") +   
 labs(title = "Distribution of Subgenres",  
 x = "Genre",  
 y = "Frequency") +  
 theme\_classic() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1, face = "bold"),  
 axis.text.y = element\_text(face = "bold"),  
 axis.title = element\_text(face = "bold"),  
 plot.title = element\_text(face = "bold")) +  
 theme(legend.position = "none")



The dataset contains 16 different subgenres, and all 32000 audio clips are distributed evenly. This balanced distribution will allow us to avoid bias in the analysis.

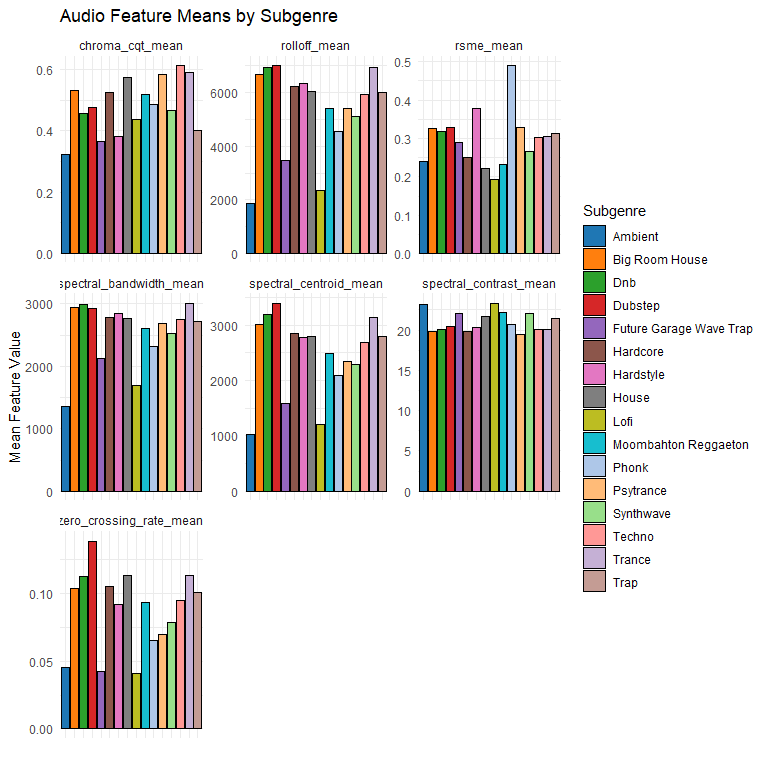
## Overview

Let us look at the means of the audio features for each subgenre to get an idea of the distribution of the data.

# The syntax below will create a table of the audio feature means for each subgenre  
  
feature\_means\_summary <- edm\_data %>%  
 group\_by(label) %>%  
 summarise(  
 rsme\_mean = mean(rmse\_mean),  
 spectral\_centroid\_mean = mean(spectral\_centroid\_mean),  
 spectral\_bandwidth\_mean = mean(spectral\_bandwidth\_mean),  
 rolloff\_mean = mean(rolloff\_mean),  
 zero\_crossing\_rate\_mean = mean(zero\_crossing\_rate\_mean),  
 chroma\_cqt\_mean = mean(chroma\_cqt\_mean),  
 spectral\_contrast\_mean = mean(spectral\_contrast\_mean)  
 )

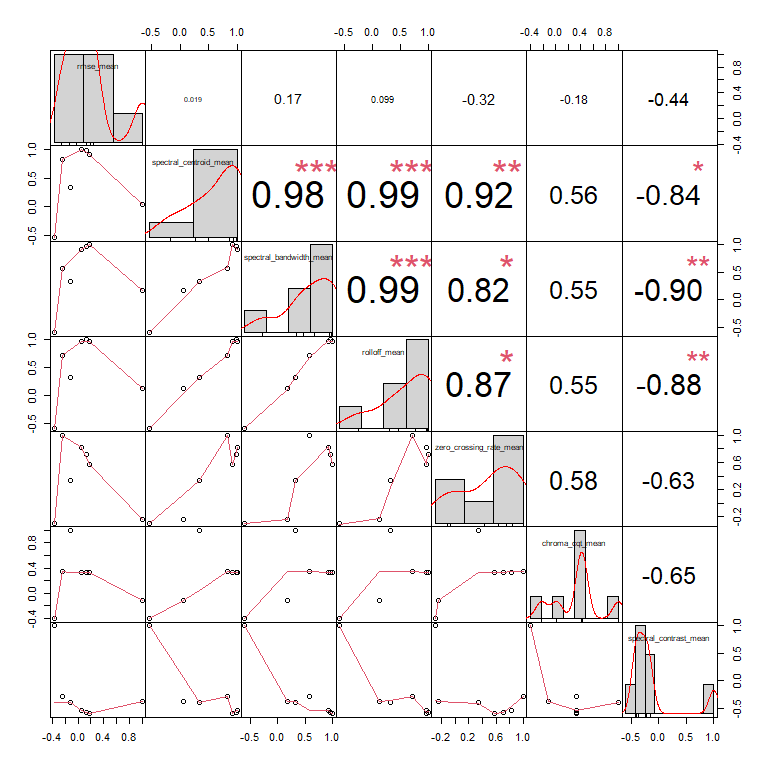
Visualizing the data will help us understand the distribution of the audio features across the different genres.

# The syntax below will create a faceted bar plot of the audio feature means for each subgenre  
  
# Reshape the data from wide to long format and format the labels  
feature\_means\_long <- feature\_means\_summary %>%  
 pivot\_longer(  
 cols = -label,  
 names\_to = "Feature",  
 values\_to = "Mean"  
 ) %>%  
 mutate(label = gsub("\_", " ", label),   
 label = tools::toTitleCase(label))   
  
ggplot(feature\_means\_long, aes(x = label, y = Mean, fill = label)) +  
 geom\_col(color = "black") +  
 scale\_fill\_paletteer\_d(palette = "ggsci::category20\_d3") +  
 facet\_wrap(~ Feature, scales = "free\_y", ncol = 3) +  
 labs(title = "Audio Feature Means by Subgenre",  
 x = "", # X-axis label removed since it's evident from the labels  
 y = "Mean Feature Value",  
 fill = "Subgenre") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_blank()) # Adjust text alignment for better readability



Now, let us see if there is any correlation between the audio features.

# The syntax below will create a correlation matrix of the audio features  
  
# Select the features to include in the correlation matrix  
selected\_features <- edm\_data %>%   
 select(rmse\_mean, spectral\_centroid\_mean, spectral\_bandwidth\_mean, rolloff\_mean,  
 zero\_crossing\_rate\_mean, chroma\_cqt\_mean, spectral\_contrast\_mean)  
  
cor\_matrix <- cor(selected\_features, use = "complete.obs")   
chart.Correlation(cor\_matrix, histogram = TRUE)

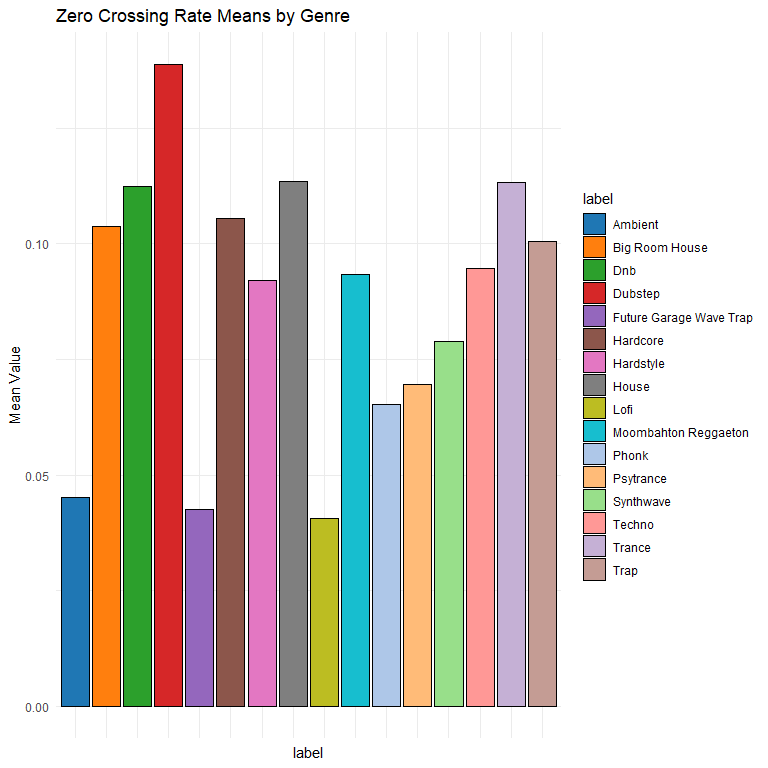


**Correlation Implications**

The very high correlations among spectral centroid, bandwidth, and rolloff suggest that these features are closely related descriptors of the frequency content of a sound. They are critical for distinguishing different types of sounds and could be key features for subgenre classification in EDM.

Since we are focusing on the “zero\_crossing\_rate\_mean” feature, let us create a separate plot for this feature to get a better view of the data.

# The syntax below will create a bar chart of the zero crossing rate means for each subgenre  
  
# Filter the data for the "zero\_crossing\_rate\_mean" feature  
zerocrossing\_feature\_data <- feature\_means\_long %>%  
 filter(Feature == "zero\_crossing\_rate\_mean")  
  
ggplot(zerocrossing\_feature\_data, aes(x = label, y = Mean, fill = label)) +  
 geom\_col(color = "black") +  
 scale\_fill\_paletteer\_d(palette = "ggsci::category20\_d3") +  
 labs(title = "Zero Crossing Rate Means by Genre",  
 y = "Mean Value") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_blank())



Viewing the data this way, we can see differences in the mean zero crossing rate values across the different genres and some relationships between the genres. For example, the “dubstep” subgenre has a much higher mean zero crossing rate compared to other genres like “ambient” and “lo-fi”. This is likely because of the percussive nature of dubstep music, which has a higher rate of zero crossings due to the sharp transitions in the waveform compared to the smoother transitions in ambient and Lofi music.

## ANOVA Test

We can perform an Analysis of Variance test to determine if significant differences exist in the zero crossing rate means among the different genres (ANOVA). ANOVA is used because it is specifically designed to compare means across three or more groups. In this case, I am examining differences in means across 16 different EDM genres.

Assumptions of this test:

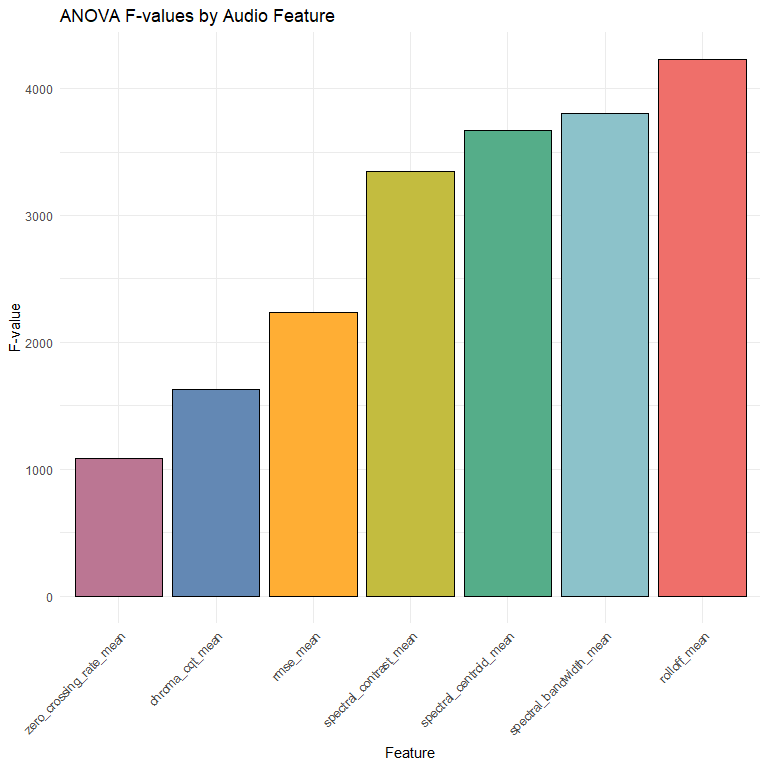
* Population distribution is normal
* Samples are random and independent
* Homogeneity of sample variance

#The syntax below will run an ANOVA test   
  
# For each feature, run ANOVA test and store results  
anova\_features <- list()  
features <- c("rmse\_mean", "spectral\_centroid\_mean", "spectral\_bandwidth\_mean",  
 "rolloff\_mean", "zero\_crossing\_rate\_mean", "chroma\_cqt\_mean", "spectral\_contrast\_mean")  
  
anova\_summaries <- data.frame(Feature = character(), F\_value = numeric(), Pr\_F = numeric(), stringsAsFactors = FALSE)  
  
for (feature in features) {  
 # Running ANOVA for each feature  
 model <- aov(as.formula(paste(feature, "~ label")), data = edm\_data)  
 anova\_features[[feature]] <- model  
  
 # Extract summary  
 summary\_model <- summary(model)  
 anova\_summaries <- rbind(anova\_summaries, data.frame(  
 Feature = feature,  
 F\_value = summary\_model[[1]]["label", "F value"],  
 Pr\_F = summary\_model[[1]]["label", "Pr(>F)"]  
 ))  
}  
  
lapply(anova\_features, summary)

## $rmse\_mean  
## Df Sum Sq Mean Sq F value Pr(>F)   
## label 15 147.8 9.852 2233 <2e-16 \*\*\*  
## Residuals 31984 141.1 0.004   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## $spectral\_centroid\_mean  
## Df Sum Sq Mean Sq F value Pr(>F)   
## label 15 1.462e+10 974769064 3667 <2e-16 \*\*\*  
## Residuals 31984 8.502e+09 265815   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## $spectral\_bandwidth\_mean  
## Df Sum Sq Mean Sq F value Pr(>F)   
## label 15 6.741e+09 449385768 3802 <2e-16 \*\*\*  
## Residuals 31984 3.780e+09 118183   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## $rolloff\_mean  
## Df Sum Sq Mean Sq F value Pr(>F)   
## label 15 7.535e+10 5.024e+09 4230 <2e-16 \*\*\*  
## Residuals 31984 3.799e+10 1.188e+06   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## $zero\_crossing\_rate\_mean  
## Df Sum Sq Mean Sq F value Pr(>F)   
## label 15 24.69 1.6461 1088 <2e-16 \*\*\*  
## Residuals 31984 48.38 0.0015   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## $chroma\_cqt\_mean  
## Df Sum Sq Mean Sq F value Pr(>F)   
## label 15 222.2 14.813 1631 <2e-16 \*\*\*  
## Residuals 31984 290.5 0.009   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## $spectral\_contrast\_mean  
## Df Sum Sq Mean Sq F value Pr(>F)   
## label 15 46267 3084.5 3347 <2e-16 \*\*\*  
## Residuals 31984 29472 0.9   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Visualizing the results of the ANOVA test.

#The syntax below will create a bar chart of the F-values for each feature  
  
# Convert p-values to negative log scale for better visualization  
anova\_summaries$neg\_log\_p <- -log10(anova\_summaries$Pr\_F)  
  
ggplot(anova\_summaries, aes(x = reorder(Feature, F\_value), y = F\_value, fill = Feature)) +  
 geom\_col(show.legend = FALSE, color = "black") +  
 scale\_fill\_paletteer\_d(palette = "ggthemes::Superfishel\_Stone") +  
 labs(title = "ANOVA F-values by Audio Feature",  
 x = "Feature",  
 y = "F-value") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))



## ANOVA Analysis

* rmse\_mean : significant difference in energy levels across genres
  + F value: 2233
  + p-value: <2e-16
* spectral\_centroid\_mean: significant differences in sound brightness across genres
  + F value: 3667
  + p-value: <2e-16
* spectral\_bandwidth\_mean: significant differences in spread of frequencies around the spectral centroid across genres
  + F value: 3802
  + p-value: <2e-16
* rolloff\_mean: significant differences in the cuttoff frequencies across genres
  + F value: 4230
  + p-value: <2e-16
* zero\_crossing\_rate\_mean: significant differences in the rate of signal changes across genres
  + F value: 1088
  + p-value: <2e-16
* chroma\_cqt\_mean: significant differences in harmonic content across genres
  + F value: 1631
  + p-value: <2e-16.
* spectral\_contrast\_mean: significant differences in dynamic range across genres
  + F value: 3347
  + p-value: <2e-16

## Answering the Research Question

The ANOVA results show that all tested audio features significantly vary across genres. Features like Spectral Rolloff, Spectral Centroid, Spectral Contrast, and Spectral Bandwidth are strong discriminators of subgenres, given their high F-values. This could make them potent tools for further subgenre classification.

The zero crossing rate has a lower F-value than other features, but it still shows significant differences across genres. This suggests that it might be a less powerful discriminator. However, it still contributes to subgenre differentiation. Due to this, I will reject the null hypothesis and accept the alternative hypothesis that the zero crossing rate is significantly different across the EDM genres in the dataset.

## Principal Component Analysis

I am not satisfied with the initial findings of this report. Since I aim to predict the subgenres accurately, I will use more features for further analysis. So far, I only used seven variables that would be the most important in determining the subgenre of the song. However, for a comprehensive analysis, I will use all 130 variables in the dataset to find a statistically relevant pattern in the data that can be used to classify the subgenres. With Principal Component Analysis (PCA), I can reduce the dimensionality of the data and identify the essential features that contribute to the variance in the dataset.

# The syntax below will perform a Principal Component Analysis (PCA) on the dataset to reduce the dimensionality of the data  
  
numeric\_data <- edm\_data %>% select(-label)  
  
scaled\_data <- scale(numeric\_data)  
  
pca\_result <- prcomp(scaled\_data, center = TRUE, scale. = TRUE)  
  
pca\_scores <- data.frame(pca\_result$x)  
  
pca\_scores$label <- edm\_data$label

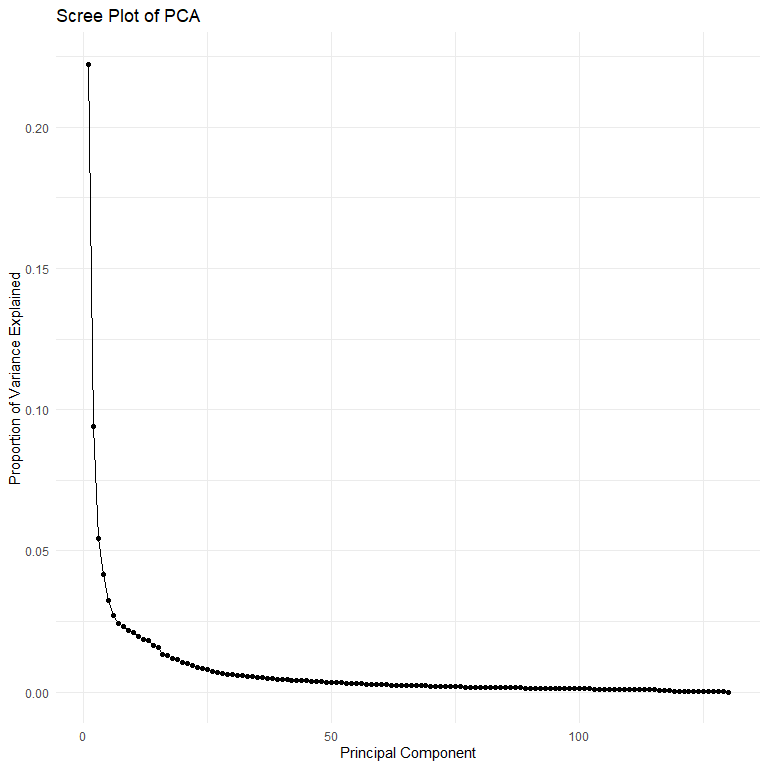
Convert the PCA results to percentages to determine the variance explained by each principal component.

# The syntax below will calculate the variance of the principal components  
  
var\_explained <- pca\_result$sdev^2 / sum(pca\_result$sdev^2) \* 100   
  
pca\_var\_df <- data.frame(  
 PC = 1:length(var\_explained),   
 Variance = var\_explained  
)  
  
print(pca\_var\_df)

## PC Variance  
## 1 1 22.241708537  
## 2 2 9.410270212  
## 3 3 5.455589363  
## 4 4 4.179667753  
## 5 5 3.242969720  
## 6 6 2.728870096  
## 7 7 2.438032108  
## 8 8 2.310069758  
## 9 9 2.187634715  
## 10 10 2.105575078  
## 11 11 1.983519539  
## 12 12 1.874906553  
## 13 13 1.839999895  
## 14 14 1.657897058  
## 15 15 1.583060503  
## 16 16 1.336399736  
## 17 17 1.297214587  
## 18 18 1.190170034  
## 19 19 1.145842973  
## 20 20 1.065266401  
## 21 21 1.009450448  
## 22 22 0.948119610  
## 23 23 0.884824402  
## 24 24 0.837927161  
## 25 25 0.808920175  
## 26 26 0.713136975  
## 27 27 0.698434796  
## 28 28 0.645414019  
## 29 29 0.617433427  
## 30 30 0.607238018  
## 31 31 0.594787480  
## 32 32 0.584694510  
## 33 33 0.563448907  
## 34 34 0.536486578  
## 35 35 0.520651267  
## 36 36 0.506975791  
## 37 37 0.488322430  
## 38 38 0.468672139  
## 39 39 0.459485535  
## 40 40 0.454173299  
## 41 41 0.435658893  
## 42 42 0.427098661  
## 43 43 0.423997632  
## 44 44 0.422574450  
## 45 45 0.407041054  
## 46 46 0.390577952  
## 47 47 0.386814952  
## 48 48 0.360549601  
## 49 49 0.356583704  
## 50 50 0.342283483  
## 51 51 0.342053487  
## 52 52 0.324077423  
## 53 53 0.320626787  
## 54 54 0.309220215  
## 55 55 0.306697564  
## 56 56 0.288921412  
## 57 57 0.283639500  
## 58 58 0.278854184  
## 59 59 0.272444759  
## 60 60 0.260874197  
## 61 61 0.256015924  
## 62 62 0.251056485  
## 63 63 0.243029898  
## 64 64 0.239513902  
## 65 65 0.235212977  
## 66 66 0.229045488  
## 67 67 0.222142967  
## 68 68 0.218925643  
## 69 69 0.217339894  
## 70 70 0.206807962  
## 71 71 0.203712137  
## 72 72 0.203334489  
## 73 73 0.201993163  
## 74 74 0.192499625  
## 75 75 0.189594104  
## 76 76 0.184675743  
## 77 77 0.181516346  
## 78 78 0.178898739  
## 79 79 0.176283036  
## 80 80 0.170793059  
## 81 81 0.165448120  
## 82 82 0.164186731  
## 83 83 0.161854447  
## 84 84 0.157840118  
## 85 85 0.155812551  
## 86 86 0.153823000  
## 87 87 0.147934173  
## 88 88 0.147024482  
## 89 89 0.144986240  
## 90 90 0.141733207  
## 91 91 0.138648211  
## 92 92 0.133667637  
## 93 93 0.132690709  
## 94 94 0.128891738  
## 95 95 0.128302014  
## 96 96 0.126875092  
## 97 97 0.125433682  
## 98 98 0.122118495  
## 99 99 0.119257082  
## 100 100 0.116959808  
## 101 101 0.113981532  
## 102 102 0.112052194  
## 103 103 0.110286931  
## 104 104 0.105412261  
## 105 105 0.104631745  
## 106 106 0.104114011  
## 107 107 0.101463043  
## 108 108 0.098096770  
## 109 109 0.096273383  
## 110 110 0.094466437  
## 111 111 0.092699143  
## 112 112 0.091755405  
## 113 113 0.087370065  
## 114 114 0.082380956  
## 115 115 0.076980818  
## 116 116 0.070678842  
## 117 117 0.049535823  
## 118 118 0.043025154  
## 119 119 0.037062458  
## 120 120 0.022335219  
## 121 121 0.021520918  
## 122 122 0.020473731  
## 123 123 0.018426474  
## 124 124 0.014268406  
## 125 125 0.012365810  
## 126 126 0.011945692  
## 127 127 0.009412960  
## 128 128 0.008505942  
## 129 129 0.006218258  
## 130 130 0.002529010

Using the results of the PCA, a scree plot can be used to observe the variance explained by each principal component.

# The syntax below will create a scree plot of the PCA results  
  
# Create a data frame for the variance explained  
var\_explained <- data.frame(PC = 1:length(pca\_result$sdev),   
 Variance = pca\_result$sdev^2 / sum(pca\_result$sdev^2))  
  
ggplot(var\_explained, aes(x = PC, y = Variance)) +  
 geom\_line() +   
 geom\_point() +  
 theme\_minimal() +  
 labs(title = "Scree Plot of PCA", x = "Principal Component", y = "Proportion of Variance Explained")



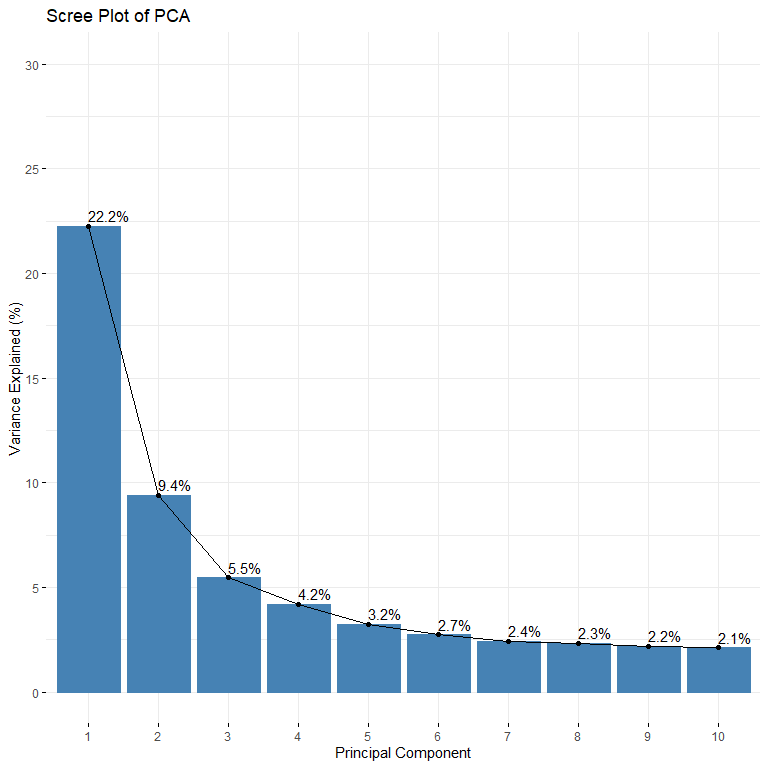
Lets calculate how many principal components are needed to explain 80%, 85%, 90%, 95%, and 99% of the variance. This will help me determine how many principal components to use in a predictive model.

# The syntax below will calculate the number of principal components needed to explain 80%, 85%, 90%, 95%, and 99% of the variance  
  
# PCA result stored in pca\_result  
pca\_result <- prcomp(scaled\_data, center = TRUE, scale. = TRUE)  
  
# Calculate the proportion of variance explained by each PC  
variance\_explained <- pca\_result$sdev^2 / sum(pca\_result$sdev^2)  
  
# Calculate cumulative variance explained  
cumulative\_variance <- cumsum(variance\_explained)  
  
# Desired levels of explained variance  
levels\_of\_variance <- c(0.8, 0.85, 0.9, 0.95, 0.99)  
  
# Determine the number of PCs required to reach each level of explained variance  
pcs\_needed <- sapply(levels\_of\_variance, function(x) which(cumulative\_variance >= x)[1])  
  
results\_df <- data.frame(  
 `Variance\_Explained` = c("80%", "85%", "90%", "95%", "99%"),  
 `Number\_of\_PCs` = pcs\_needed  
)  
  
results\_df

## Variance\_Explained Number\_of\_PCs  
## 1 80% 32  
## 2 85% 42  
## 3 90% 56  
## 4 95% 78  
## 5 99% 108

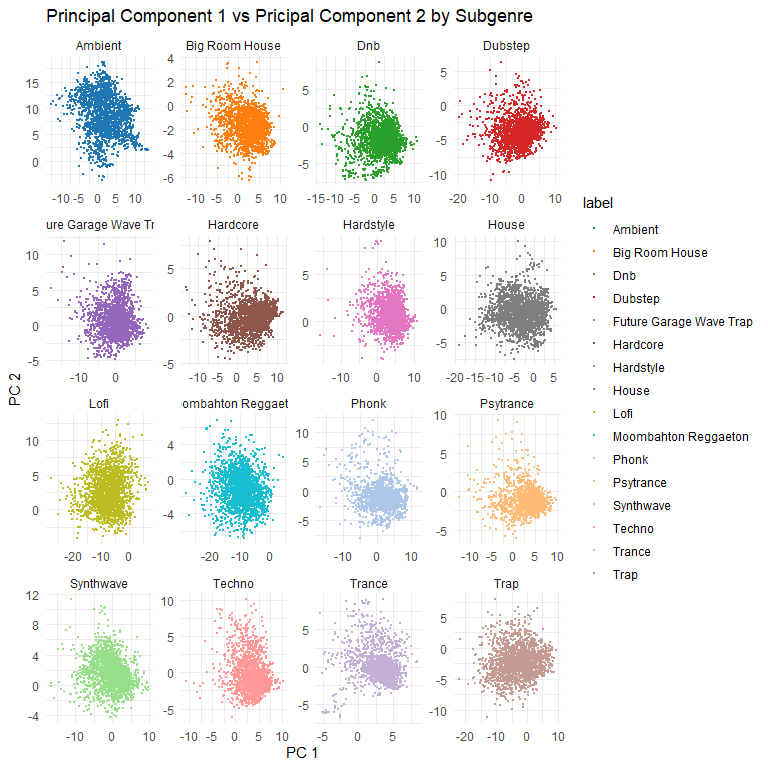
Out of the 130 principal components, the first ten explain 56.30039% of the variance in the data.

# The syntax below will create a scree plot of the PCA results  
  
fviz\_eig(pca\_result, addlabels = TRUE, ylim = c(0, 30)) +  
 scale\_y\_continuous(breaks = seq(0, 30, by = 5)) +  
 labs(title = "Scree Plot of PCA",  
 x = "Principal Component",  
 y = "Variance Explained (%)")

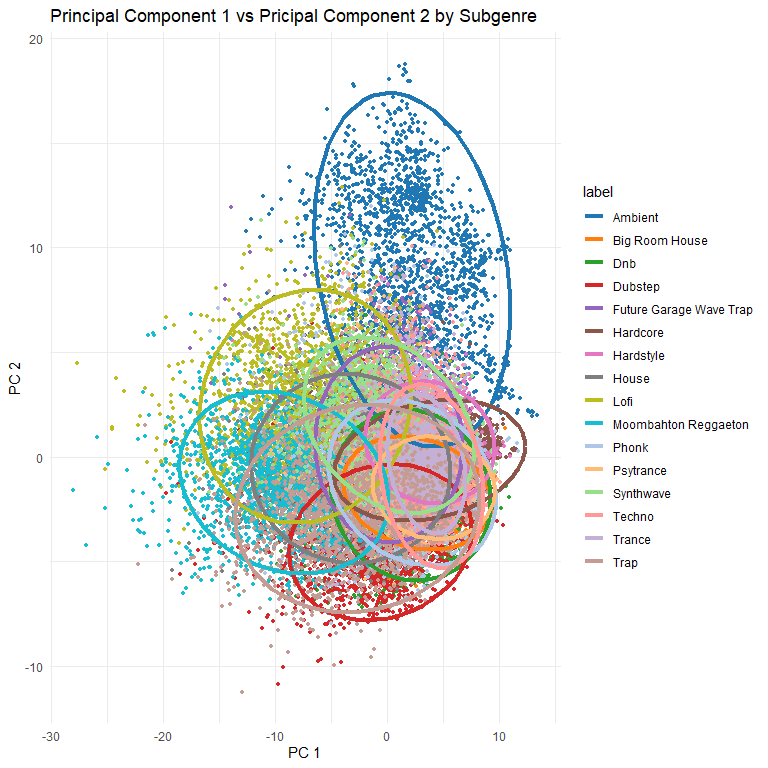


# The syntax below will extract the first 5 principal components and combine them with the original EDM data  
  
pc\_scores <- pca\_result$x[, 1:5]  
  
edm\_data\_with\_pcs <- cbind(edm\_data, pc\_scores)

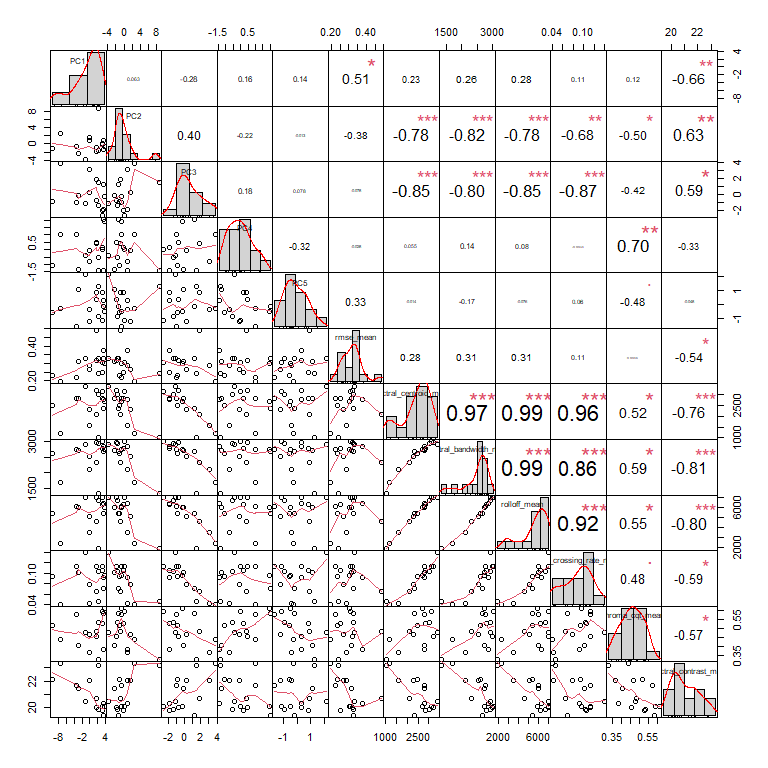
# The syntax below will create a scatter plot of the first two principal components with labels wrapped  
  
# Edit the label names  
edm\_data\_with\_pcs <- edm\_data\_with\_pcs %>%  
 mutate(label = gsub("\_", " ", label),   
 label = toTitleCase(label))   
  
ggplot(edm\_data\_with\_pcs, aes(x = PC1, y = PC2, color = label)) +  
 geom\_point(size = 0.5) +   
 scale\_color\_paletteer\_d(palette = "ggsci::category20\_d3") +  
 labs(title = "Principal Component 1 vs Pricipal Component 2 by Subgenre", x = "PC 1", y = "PC 2") +  
 theme\_minimal() +  
 facet\_wrap(~ label, scales = "free")



# The syntax below will create a scatter plot of the first two principal components with all labels  
  
# Plotting PC1 vs PC2  
ggplot(edm\_data\_with\_pcs, aes(x = PC1, y = PC2, color = label)) +  
 geom\_point(size = 1) +   
 theme\_minimal() +   
 scale\_color\_paletteer\_d(palette = "ggsci::category20\_d3") +  
 stat\_ellipse(size = 1.5) +   
 labs(title = "Principal Component 1 vs Pricipal Component 2 by Subgenre", x = "PC 1", y = "PC 2")



# The syntax below will create a correlation matrix of the first 5 principal components and the means of the initial 7 audio features  
  
edm\_data\_mean <- edm\_data\_with\_pcs %>%  
 group\_by(label) %>%   
 summarise(across(where(is.numeric), mean, na.rm = TRUE))  
  
  
selected\_columns <- c("PC1", "PC2", "PC3", "PC4", "PC5","rmse\_mean", "spectral\_centroid\_mean", "spectral\_bandwidth\_mean",  
 "rolloff\_mean", "zero\_crossing\_rate\_mean", "chroma\_cqt\_mean", "spectral\_contrast\_mean" )   
chart.Correlation(edm\_data\_mean[selected\_columns], histogram = TRUE)



## Conclusion

The PCA results show that this data is robust enough to predict subgenres. For the future of this project, machine learning techniques such as K-means clustering will be used. A potential use case for a model would be for advancing music recommendation systems. Recommendation algorithms could suggest songs that align more closely with listeners’ preferences, identified through cluster membership.

## Citations

Bambooforest. (n.d.). Bambooforest/APY313: Data Science of Culture and language. GitHub. <https://github.com/bambooforest/APY313/tree/main>

Hvitfeldt E. (2021). paletteer: Comprehensive Collection of Color Palettes. version 1.3.0. <https://github.com/EmilHvitfeldt/paletteer>

Kassambara A (2023). *ggpubr: ‘ggplot2’ Based Publication Ready Plots*. R package version 0.6.0, <https://CRAN.R-project.org/package=ggpubr>.

Kassambara A, Mundt F (2020). *factoextra: Extract and Visualize the Results of Multivariate Data Analyses*. R package version 1.0.7, <https://CRAN.R-project.org/package=factoextra>.

Lafs. (2023, February 20). A brief history of EDM. The Los Angeles Film School. <https://www.lafilm.edu/blog/brief-history-edm/>

official, S. (2024, April 20). EDM music genres. Kaggle. <https://www.kaggle.com/datasets/sivadithiyan/edm-music-genres>

Peterson BG, Carl P (2020). *PerformanceAnalytics: Econometric Tools for Performance and Risk Analysis*. R package version 2.0.4, <https://CRAN.R-project.org/package=PerformanceAnalytics>.

Hvitfeldt E. (2021). paletteer: Comprehensive Collection of Color Palettes. version 1.3.0. <https://github.com/EmilHvitfeldt/paletteer>

Wickham H, Averick M, Bryan J, Chang W, McGowan LD, François R, Grolemund G, Hayes A, Henry L, Hester J, Kuhn M, Pedersen TL, Miller E, Bache SM, Müller K, Ooms J, Robinson D, Seidel DP, Spinu V, Takahashi K, Vaughan D, Wilke C, Woo K, Yutani H (2019). “Welcome to the tidyverse.” *Journal of Open Source Software*, *4*(43), 1686. <doi:10.21105/joss.01686> <https://doi.org/10.21105/joss.01686>.

Xie Y (2024). *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.46, <https://yihui.org/knitr/>.