```
#Loading Kaggle dataset
library(dplyr)
full <- read.csv("/Users/shreyakusumanchi/Downloads/spotify-2023.csv")
full
View(full)

#Check dimensions and column names
names(full)
dim(full)

#Cleaning dataset by removing columns and converting all values to numeric
columns_to_remove <- c(2,6,12,13,14)  # Index of columns to remove
spotify_full <- full[, -columns_to_remove]
typeof(spotify_full$streams)
spotify_full$streams = as.numeric(spotify_full$streams)
spotify_full$streams
spotify_full2 <- spotify_full %>%
  filter(streams > 1e9)
dim(spotify_full2)

#Descriptive statistics of the entire dataset
summary(spotify_full2$streams)
View(spotify_full2)
min(spotify_full2$streams)
max(spotify_full2$streams)



#Plot 1: Valence vs. Streams with a regression line & Correlation Coefficient
plot(spotify_full2$valence_., spotify_full2$streams,
     main = "Valence vs. Streams", xlab = "Valence Score", ylab = "Number of Streams")
fit_valence <- lm(spotify_full2$streams ~ spotify_full2$valence_.)
abline(fit_valence, col = "orange")
cor(spotify_full2$valence_., spotify_full2$streams)

#Linear model with p-value for single attribute
valence <- lm(streams ~ valence_., data = spotify_full2)
valence
summary(valence)

#Descriptive statistics
summary(spotify_full2$valence_.)
mean(spotify_full2$valence_.)
```

#Plot 2: Energy vs. Streams with a regression line & Correlation Coefficient

```
plot(spotify_full2$energy_., spotify_full2$streams,
    main = "Energy vs. Streams",  xlab = "Energy Score", ylab = "Number of Streams")
fit_energy <- lm(spotify_full2$streams ~ spotify_full2$energy_.)
abline(fit_energy, col = "red")
cor(spotify_full2$energy_., spotify_full2$streams)
```

#Linear model with p-value for single attribute

```
energy <- lm(streams ~ energy_., data = spotify_full2)
energy
summary(energy)
```

#Descriptive statistics

```
summary(spotify_full2$energy_.)
mean(spotify_full2$energy_.)
```

 # Plot 3: BPM vs. Streams with a regression line & Correlation Coefficient

```
plot(spotify_full2$bpm, spotify_full2$streams,
    main = "BPM vs. Streams",  xlab = "Beats Per Minute", ylab = "Number of Streams")
fit_bpm <- lm(spotify_full2$streams ~ spotify_full2$bpm)
abline(fit_bpm, col = "green")
cor(spotify_full2$bpm, spotify_full2$streams)
```

#Linear model with p-value for single attribute

```
bpm <- lm(streams ~ bpm, data = spotify_full2)
bpm
summary(bpm)
```

#Descriptive statistics

```
summary(spotify_full2$bpm)
mean(spotify_full2$bpm)
```

 # Plot 4: Instrumentalness vs. Streams with a regression line & Correlation Coefficient

```
plot(spotify_full2$instrumentalness_., spotify_full2$streams,
    main = "Instrumentalness vs. Streams", xlab = "Instrumentalness Score", ylab = "Number of Streams")
fit_instrumentalness <- lm(spotify_full2$streams ~ spotify_full2$instrumentalness_.)
abline(fit_instrumentalness, col = "purple")
cor(spotify_full2$instrumentalness_., spotify_full2$streams)
```

```r
#Linear model with p-value for single attribute
instrument <- lm(streams ~ instrumentalness_., data = spotify_full2)
instrument
summary(instrument)

#Descriptive statistics
summary(spotify_full2$instrumentalness_.)
mean(spotify_full2$instrumentalness_.)



# Plot 5: Danceability vs. Streams with a regression line & Correlation Coefficient
plot(spotify_full2$danceability_., spotify_full2$streams,
     main = "Danceability vs. Streams",  xlab = "Danceability Score", ylab = "Number of Streams")
fit_danceability <- lm(spotify_full2$streams ~ spotify_full2$danceability_.)
abline(fit_danceability, col = "orange")
cor(spotify_full2$danceability_., spotify_full2$streams)

#Linear model with p-value for single attribute
dance <- lm(streams ~ danceability_., data = spotify_full2)
dance
summary(dance)

#Descriptive statistics
summary(spotify_full2$danceability_.)
mean(spotify_full2$danceability_.)



# Plot 6: Speechiness vs. Streams with a regression line & Correlation Coefficient
plot(spotify_full2$speechiness_., spotify_full2$streams,
     main = "Speechiness vs. Streams", xlab = "Speechiness Score", ylab = "Number of Streams")
fit_Speechiness <- lm(spotify_full2$streams ~ spotify_full2$speechiness_.)
abline(fit_Speechiness, col = "black")
cor(spotify_full2$speechiness_., spotify_full2$streams)

#Linear model with p-value for single attribute
speech <- lm(streams ~ speechiness_., data = spotify_full2)
speech
summary(speech)

#Descriptive statistics
summary(spotify_full2$speechiness_.)
```

```
mean(spotify_full2$speechiness_.)
```

 # Plot 7: Acousticness vs. Streams with a regression line & Correlation Coefficient

```
plot(spotify_full2$acousticness_., spotify_full2$streams,
    main = "Acousticness vs. Streams", xlab = "Acousticness Score", ylab = "Number of Streams")
fit_Acousticness <- lm(spotify_full2$streams ~ spotify_full2$acousticness_.)
abline(fit_Acousticness, col = "yellow")
cor(spotify_full2$acousticness_., spotify_full2$streams)
```

#Linear model with p-value for single attribute

```
Acousticness <- lm(streams ~ acousticness_., data = spotify_full2)
Acousticness
summary(Acousticness)
```

#Descriptive statistics

```
summary(spotify_full2$Acousticness)
mean(spotify_full2$Acousticness)
```

 # Plot 8: Liveness vs. Streams with a regression line & Correlation Coefficient

```
plot(spotify_full2$liveness_., spotify_full2$streams,
    main = "Liveness vs. Streams", xlab = "Liveness Score", ylab = "Number of Streams")
fit_Liveness <- lm(spotify_full2$streams ~ spotify_full2$liveness_.)
abline(fit_Liveness, col = "pink")
cor(spotify_full2$liveness_., spotify_full2$streams)
```

#Linear model with p-value for single attribute

```
Liveness <- lm(streams ~ liveness_., data = spotify_full2)
Liveness
summary(Liveness)
```

#Descriptive statistics

```
summary(spotify_full2$liveness_.)
mean(spotify_full2$liveness_.)
```

#Model with all musical attributes

```
model <- lm(streams ~ valence_. + energy_. + bpm + danceability_. + speechiness_. + acousticness_. +
liveness_., data = spotify_full2)
model
summary(model)
```

```
fit_model <- lm(streams ~ valence_. + energy_. + bpm + danceability_. + speechiness_. + acousticness_.
+ liveness_., data = spotify_full2)
abline(fit_model, col = "purple")

#Model with select musical attributes
model2 <- lm(streams ~ valence_. + energy_. + bpm + danceability_. + speechiness_., data =
spotify_full2)
model2
summary(model2)
fit_model2 <- lm(streams ~ valence_. + energy_. + bpm + danceability_. + speechiness_., data =
spotify_full2)
abline(fit_model2, col = "purple")
plot(model2$fitted.values, residuals(model2),
    xlab = "Fitted values", ylab = "Residuals",
    main = "Residual plot")
abline(h = 0, col = "purple")




install.packages("glmnet")
install.packages("glmnetUtils")

library(glmnetUtils)
library(glmnet)


# Extracting predictors and response variable
predictors <- spotify_full2[, c("valence_.", "energy_.", "bpm", "danceability_.", "speechiness_.",
"acousticness_.", "liveness_.")]
response <- spotify_full2$streams
predictors <- scale(predictors)

# Fit Lasso regression model
lasso_model <- cv.glmnet(as.matrix(predictors), response, alpha = 1)

# Get the best lambda value selected by cross-validation
best_lambda <- lasso_model$lambda.min

# Extract the coefficients for the best lambda
best_coef <- coef(lasso_model, s = best_lambda)
best_coef

# Select predictors with non-zero coefficients
```

selected_predictors <- names(best_coef)[best_coef != 0]
selected_predictors

plot(spotify_full2$valence_., spotify_full2$streams,
    main = "Valence vs. Streams", xlab = "Valence Score", ylab = "Number of Streams")

# Fit linear regression models
fit_valence <- lm(spotify_full2$streams ~ spotify_full2$valence_.)
fit_model2 <- lm(streams ~ valence_. + energy_. + bpm + danceability_. + speechiness_., data = spotify_full2)

# Add linear regression lines with correct colors
abline(fit_valence, col = "orange")
abline(fit_model2, col = "purple")

# Add a legend
legend("topright", legend = c("Valence Line", "Model2 Line"), col = c("orange", "purple"), lty = 1, lwd = 2)

---Linear Model---

full <- read.csv("/Users/davidattar/Desktop/spotify-2023.csv")

columns_to_remove <- c (1,2,3,4,5,6,7,8,10,11,12,13,14,15,16,17) # Index of columns to remove, removing non-numeric / Unused variables

undesirables <- full[, -columns_to_remove]

data <- undesirables

data$streams <- as.integer(gsub("[^0-9]", "", data$streams)) #Changing the values of the stream column from characters to integers

Selecting features and target variable

var_test <- c("danceability_.", "valence_.", "energy_.", "acousticness_.", "instrumentalness_.", "liveness_.", "speechiness_.")

target <- "streams"

Split the dataset into training and testing sets

set.seed(1) # For reproducibility

train_indices <- sample(1:nrow(data), 0.8 * nrow(data)) #Setting test + Train data to 80/20 ratio

train_data <- data[train_indices,]

```
test_data <- data[-train_indices, ]
any(is.na(train_data$streams)) #Making sure the variables all exist and are the right class
sapply(train_data, class)

train_data <- train_data[!is.na(train_data$streams), ] # Removing the na values from data$streams



model <- lm(streams ~ ., data = train_data) # Create a linear regression model
summary(model) #Summarize the model

# FOR PLOT

library(ggplot2)

# Create a data frame for the actual and predicted values
plot_data <- data.frame(Actual = test_data$streams, Predicted = predictions)

# Create a scatter plot using ggplot2 with Spotify colors
ggplot(plot_data, aes(x = Actual, y = Predicted))
  geom_point(color = "#1ED760", size = 3) +  # Spotify green
  geom_abline(intercept = 0, slope = 1, color = "#FF10F0", linetype = "dashed") +  # Spotify neon pink
dashed line
  labs(title = "Actual vs. Predicted Values",
        x = "Actual Values",
        y = "Predicted Values") +
  theme_minimal() +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank()) +  # Remove gridlines
  scale_color_manual(values = "#1ED760")
```