# Data Royale DIKE 2023

*Multiclass Prediction Prediksi genre musik berdasarkan fitur*

**OTW OTI** 😘😘

Bagus Cipta Pratama (23/516539/PA/22097)

Hafid Sasayuda Ambardi (23/519841/PA/22308)

Joseph Greffen Komala (23/519909/PA/22318)

**DEPARTEMEN ILMU KOMPUTER DAN ELEKTRONIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS GADJAH MADA**

# 1. Pendahuluan

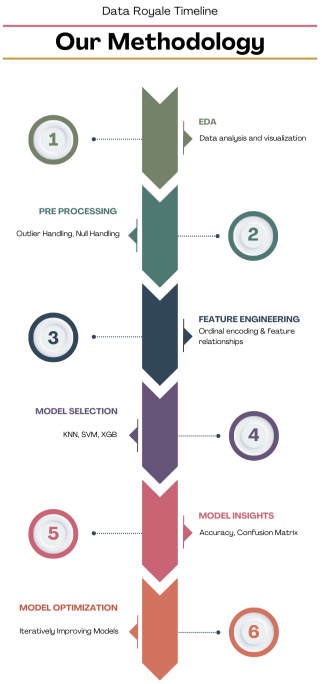
Dalam era digital yang semakin berkembang, industri musik telah mengalami transformasi yang signifikan. Penggemar musik terus meningkat, sehingga sejumlah artis musik berlomba-lomba untuk mendapatkan perhatian pendengar. Selera musik, trend, genre, dan kelompok musik populer dapat berubah sehingga industri musik harus beradaptasi dengan cepat.

Secara singkat, riset ini akan mengeksplorasi bagaimana kami dapat mengkategorikan musik berdasarkan *class*, yang tentu saja akan memudahkan penggemar musik untuk mencari musik sesuai selera mereka. Selain itu, dengan pemahaman yang lebih baik mengenai hubungan fitur musik dengan *class* para pelaku industri musik dapat mengambil keputusan yang lebih cerdas, menciptakan musik yang lebih relevan, dan merancang strategi pemasaran yang lebih efektif.

# 2. Metodologi

## a. Grafik Usulan Pendekatan

Pendekatan yang kami lakukan adalah pertama-tama melakukan analisis data eksploratif (EDA). Setelah kami mengekstrapolasi sejumlah informasi yang digunakan, kami akan beralih ke pra-pemrosesan, rekayasa fitur, pemilihan model, mengekstraksi wawasan model, dan mengoptimalkan model prediksi kami.




## b. Exploratory Data Analysis (EDA)

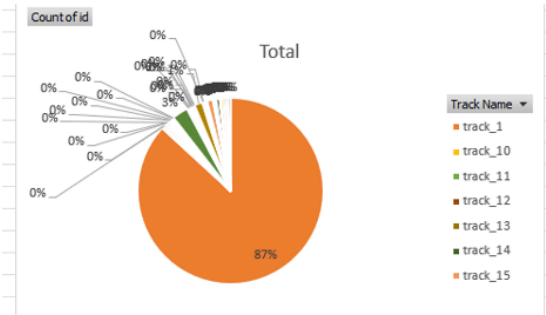
Analisis Data Eksplorasi (EDA) adalah fase awal dalam menganalisis data. Hal ini melibatkan pemeriksaan komprehensif terhadap kumpulan data

untuk menemukan data yang salah, potensi outlier, dan tren yang tetap. Melalui proses ini, EDA bertujuan untuk memberi arahan dalam pengambilan keputusan terkait pendekatan analitis dan strategi pemodelan selanjutnya. Dengan demikian, kita mulai dengan memahami dan menganalisis data.

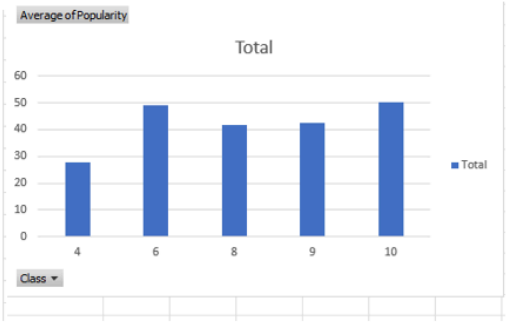
Perhatikan bahwa dalam kolom "artist name", terdapat 5265 artist yang berbeda, sehingga dapat kita simpulkan bahwa hanya sedikit lagu yang memiliki artis yang sama. Selain itu, terdapat beberapa value yang sangat mirip seperti dibawah ini:

| | |
|---|---|
| 3 | BaBos |
| 4 | BaBosc |
| 5 | BaBosck |
| 6 | BaBoscks |
| 7 | BaBosckst |
| 8 | BaBosckstr |
| 9 | BaBosckstre |
| 0 | BaBosckstree |
| 1 | BaBosckstreet |

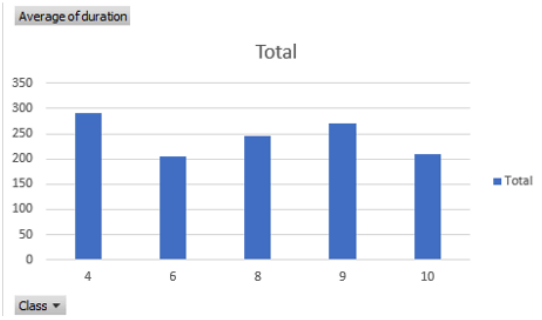
Di riset ini, kami anggap artis-artis di atas merupakan artis yang berbeda, oleh karena itu kita memilih untuk menghapus kolom nama artis, karena tidak memberikan data yang bermanfaat.



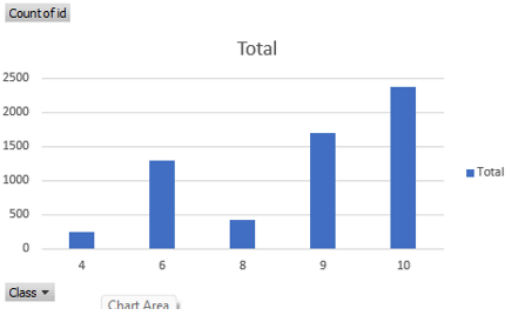

gambar 1. kolom trackname




gambar 2. Average of popularity



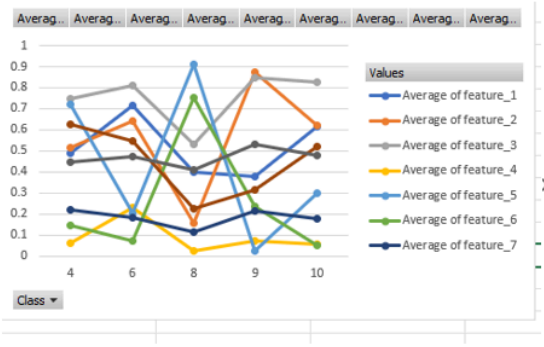

gambar 3. average of duration



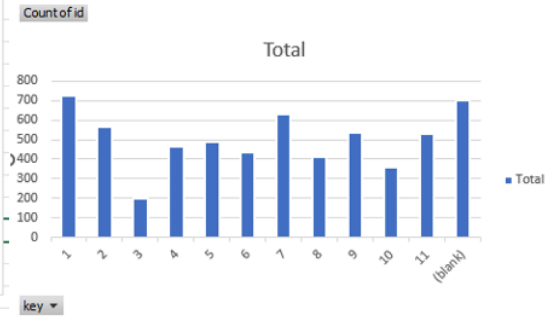

gambar 4. class

Apabila kita melihat kolom trackname, terdapat ketidakseimbangan dalam persebaran *value* di kolom ini. Ada lebih dari 85% dari data memiliki track_1 sebagai *value* dari track name nya. Lalu apabila kita melihat rata-rata nilai pada kolom "popularity" setiap *value* class terlihat bahwa tidak terjadi

ketidakseimbangan yang yang signifikan dalam data ini . Apabila kita melihat kolom “class”, terjadi ketidakseimbangan dalam data ini . Mengenai persebaran class, ada perbedaan yang cukup banyak antara jumlah data yang memiliki nilai “Class”=4 dengan jumlah data yang memiliki nilai “Class” =10, Oleh sebab itu perlu bagi kita mengatasi ketidakseimbangan ini, sebab ini bisa menimbulkan bias pada model, dan ditambah lagi kolom “Class” merupakan kolom yang menjadi target kita .

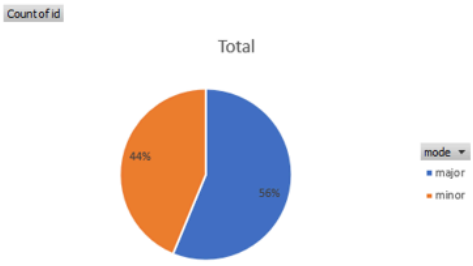



gambar 5. Feature

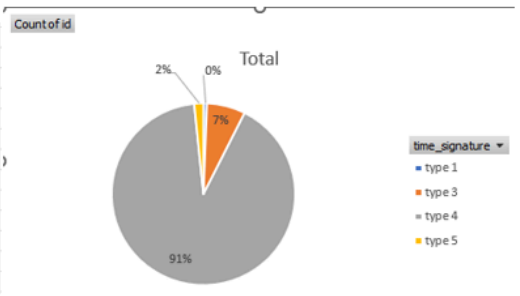



gambar 6. key

Apabila kita melihat kolom key, dalam kolom key hanya terdapat 11 *unique value*, dan jika kita melihat visualisasi dari data, memang ada perbedaan yang cukup signifikan dari jumlah data dengan value key=3 dan jumlah data dengan value key=1, tetapi apabila melihat data-data yang lain persebaran ini sudah cukup seimbang sehingga kita tidak perlu melakukan apapun.




gambar 7. mode




gambar 8. Time_signature

Selanjutnya saatnya melihat kolom “mode”. Dari grafik diatas terlihat bahwa jumlah data yang memiliki nilai “mode” sama dengan “minor” tidak jauh berbeda dengan jumlah data yang memiliki nilai “mayor” sehingga bisa dikatakan bahwa persebaran data sudah cukup baik. Dan untuk kolom terakhir mari kita melihat kolom “time_signature”. Dari penjelasan di grafik terlihat bahwa lebih dari 90% data memiliki value “type 4” pada kolom “time_signature”.

Untuk menghindari terjadinya bias pada kolom kolom yang memiliki persebaraan data tidak seimbang, maka kita perlu melakukan sesuatu terhadap kolom tersebut.

## c. Pre-Processing

### i. Outlier Handling

Dalam data, kehadiran *outlier* bisa membuat model bertindak tidak konsisten. Dalam kasus kita ada 2 kolom, yaitu "duration" dan "Popularity" yang memungkinkan memiliki *outlier* didalamnya. Kita membandingkan 2 metode dalam mengatasi *outlier* di kolom tersebut. Kita membandingkannya untuk mencari metode mana yang memiliki akurasi yang paling baik. Metode yang kita gunakan yaitu:

- **Pembatasan dengan iqr detection**

Pertama-tama kita mengelompokkan data sesuai dengan kelas yang dimilikinya. Lalu di setiap kelompok data kita, kita menerapkan pembatasan dengan metode iqr pada kolom "duration" dan "Popularity". Kemudian , untuk setiap kolom dan kelompok data, kita mencari nilai kuartil 1(Q1) dan kuartil 3(Q3), setelah itu menyimpan nilai IQR, yang bernilai sama dengan Q3-Q1. Setelah itu kita akan membuat batas atas yang nilainya 1.5*IQR+Q3 dan batas bawah yaitu,Q1-1.5*IQR. Setelah proses tersebut , apabila kita menemukan data yang memiliki nilai diatas dari batas atas maka kita akan mengubahnya menjadi sama dengan batas atas dan apabila data memiliki nilai kurang dari batas bawah, maka kita akan mengubahnya menjadi sama dengan batas bawah

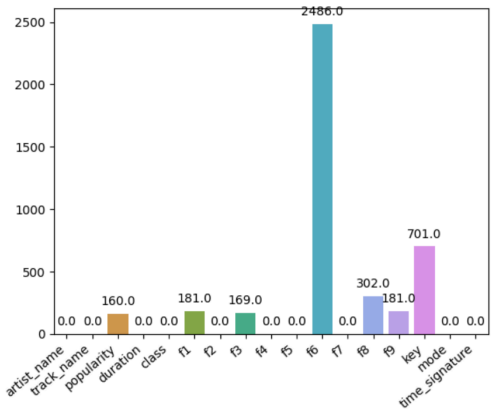
- **Pembatasan dengan z-score detection**

Pertama-tama kita mengelompokkan data sesuai dengan kelas yang dimilikinya. Lalu di setiap kelompok data kita, kita menerapkan pembatasan dengan metode iqr pada kolom "duration" dan "Popularity". Kemudian , untuk setiap kolom dan kelompok data, kita mencari standar deviasi($\boldsymbol{\sigma}$) dan rata rata nilai(μ). Lalu untuk setiap value, kita menghitung nilai z-score nya dengan formula: $z = \frac{x-\mu}{\sigma}$ Setelah proses tersebut, apabila kita menemukan data yang memiliki nilai z-score diatas 3 dari batas atas maka kita akan mengubahnya menjadi sama dengan $3 \times \sigma + \mu$ dan apabila data memiliki nilai z-score kurang dari -3, maka kita akan mengubahnya menjadi sama dengan $3 \times \sigma + \mu$ . alasan menggunakan 3 dan -3 sebagai pembatas adalah karena itu cukup sering digunakan dan menjadi standar untuk *outlier detection* dalam metode z-score

Setelah membandingkan 2 metode tersebut, kita menemukan bahwa metode yang menghasilkan akurasi terbaik adalah metode "pembatasan dengan iqr", sehingga kita memutuskan untuk menggunakan metode tersebut. Berikut adalah tabel perbandingan yang berisi akurasi dengan menggunakan metode "iqr" dan "z-score".

| IQR | Z-SCORE |
|---|---|
| 0.9331593730112513 | 0.9317745088565008 |

## ii. Null Handling




Setelah menangani data outlier , kita beralih ke *null values* .

Pengamatan lebih lanjut terhadap jumlah *null values* per fitur menunjukan kepada kita bahwa banyak fitur mengandung *null values* . selain itu , fitur ”f6” dan “key” sangat menonjol karena banyaknya *missing values* , masing-masing 2486 dan 701.

Menyadari hal ini, kami memutuskan untuk mengambil pendekatan menggunakan KNN Imputer. Kami memilih imputer KNN karena KNN memperhitungkan dengan mengambil rata-rata kumpulan data yang paling mirip dengannya. Metode ini jauh lebih baik dibandingkan metode lain seperti Iterative Imputer dan Simple Imputer karena tidak mengisi data dengan data yang sangan berbeda dengan data yang sudah ada, dan dapat mempertahankan tren di seluruh data, sehingga semakin meningkatkan keakuratan model kami.

| N Neighbors | Accuracy Increase |
|---|---|
| 5 | 0 |
| 10 | 0.00051285 |
| 15 | 0.00223473 |
| 30 | 0.00144384 |
| 50 | 0.00189335 |

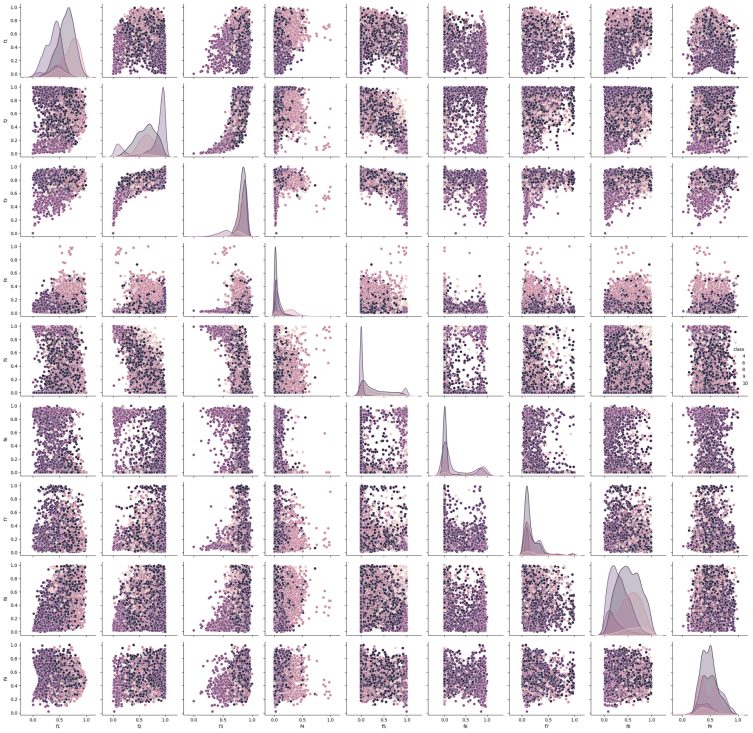
Dengan menggunakan akurasi terendah sebagai basis, kita dapat melihat bahwa ketika kita menetapkan 15 sebagai parameter untuk n_neighbours, hal ini menghasilkan peningkatan akurasi tertinggi. Oleh karena itu, kami melanjutkan untuk menangani nilai null melalui imputasi KNN dengan 15 yang ditetapkan untuk parameter n_neighbours.

## iii. Potensi kesalahan data lainnya

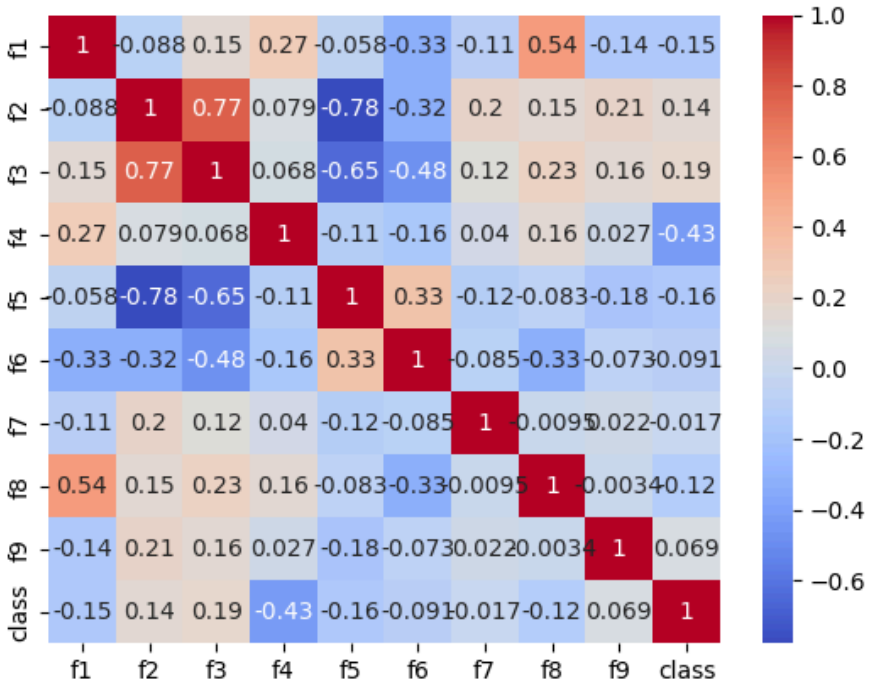
Kami juga mencari nilai float yang salah dalam data bilangan bulat , baris yang memiliki duplikasi , dan kesalahan dalam data kategorikal , namun semua data yang dikumpulkan bebas dari masalah ini .

## iv. Hubungan antar fitur

Mengidentifikasi hubungan fitur adalah bagian penting dari EDA karena membantu kita untuk mengungkap pola yang mungkin terlewatkan oleh model kami karena ini menjadi dasar untuk melakukan *fiture engineering* . untuk lebih jelasnya , lihatlah grafik visual dibawah ini .



Grafik diatas merupakan matriks dari *scatter plots* yang menunjukan hubungan suatu fitur dengan dengan fitur lainnya . Pada grafik di atas, sumbu X dan Y merepresentasikan feature_1 sampai dengan feature_9. Warna di setiap titik data pada grafik tersebut berdasarkan class. Jika kita memperhatikan grafiknya, kita dapat melihat mayoritas *feature* tidak memiliki hubungan erat dengan *feature* lain. Tetapi, juga terdapat beberapa pengecualian, seperti pada hubungan antara feature_2 dan feature_3 dan juga hubungan antara feature_2 dan feature_5.




Kita bisa mengukur hubungan satu fitur dengan fitur yang lain dengan menggunakan rumus *Pearson correlation coefficient*

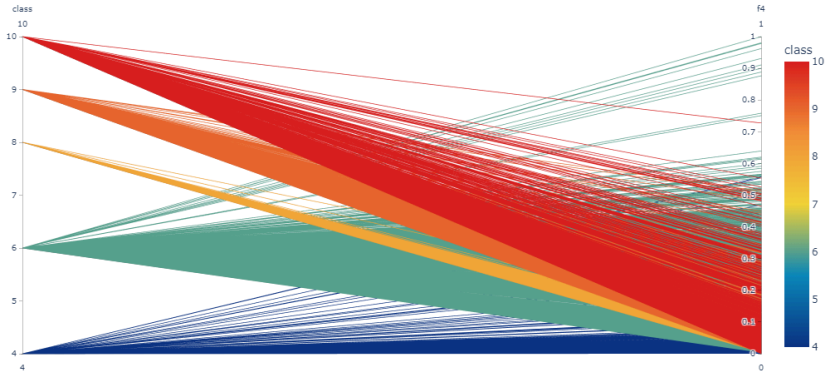
$$r = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\Sigma(x_i - \bar{x})^2 \Sigma(y_i - \bar{y})^2}}$$

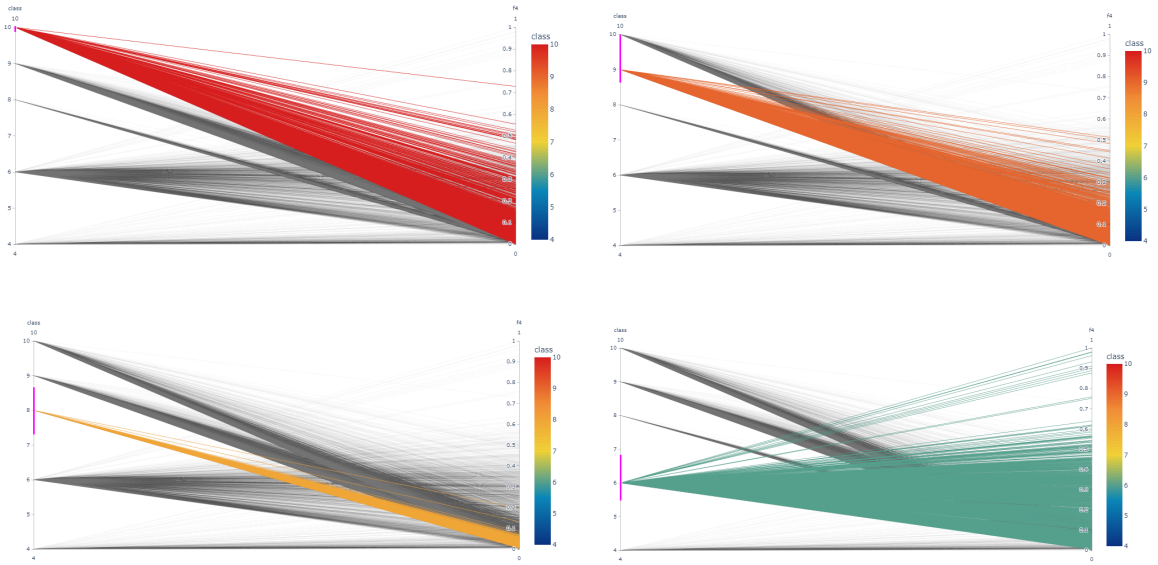
Keterangan:

- r = koefisien korelasi
- $x_i$=nilai variabel x dalam sampel
- $\bar{x}$ = rata-rata nilai variabel x
- $y_i$ = nilai variabel y dalam sampel
- $\bar{y}$ = rata-rata nilai variabel y

Selain itu, kita juga bisa melihat fitur mana yang secara langsung mempengaruhi "class", dimana fitur 4 memiliki hubungan signifikan dengan koefisien korelasi -0.43.

Untuk menganalisis korelasi f4 dengan class lebih lanjut, bisa menggunakan parallel *parallel coordinates plot* seperti pada grafik-grafik dibawah:



Pada grafik-grafik di atas, warna setiap *line segment* berdasarkan nilai class. Dengan melihat grafik, kita dapat melihat bahwa kelas 6 memiliki nilai puncak tertinggi untuk f4 , apalagi kelas 8 memiliki puncak terendah . penting untuk mengidentifikasi hubungan ini karena kita dapat melihat secara mendalam bagaimana fitur-fitur ini mempengaruhi model prediktif , dan memberi kita wawasan tentang fitur mana yang mungkin perlu kita skalakan .

| Nama Fitur | Rata-rata korelasi ($\bar{r}$) |
|---|---|
| f1 | 0.2985361300169201 |
| f2 | 0.39999445426268004 |
| f3 | 0.40227394225647134 |
| f4 | 0.21327299696795654 |
| f5 | 0.36738846484375315 |
| f6 | 0.34563547314342247 |
| f7 | 0.1893482174716678 |
| f8 | 0.2783089076245692 |
| f9 | 0.20248739310206187 |

$$\bar{r} = \frac{\Sigma|r_i|}{i}$$

Keterangan:

- r = koefisien korelasi
- $\bar{r}$=rata-rata nilai variabel r

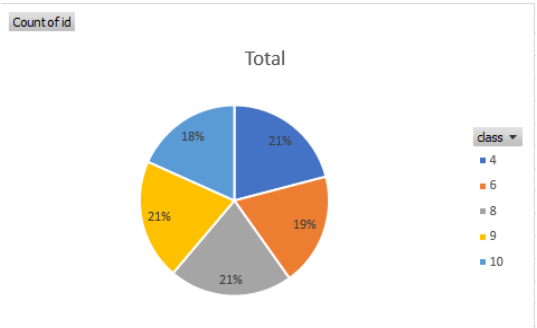
Ketika kita melihat rata-rata koefisien korelasi setiap fitur dengan fitur lain, kita bisa melihat hubungan erat satu fitur dengan fitur yang lain. Dengan seperti ini, hal tersebut menjadi bahan pertimbangan fitur mana yang berpotensi meningkatkan akurasi model prediksi, dan mana yang mengurangi serta perlu dihilangkan.

### v. Handling Imbalanced Data

Telah dijelaskan diatas bahwa persebaran data Class tidaklah merata. Adanya ketidakmerataan ini bisa menyebabkan model tidak bekerja dengan baik dan bahkan bisa menyebabkan bias pada model. Untuk mengatasinya kita perlu membuat data menjadi balanced.

Kita menggunakan kombinasi teknik *upsampling* dan *downsampling* pada data, secara spesifik kita menggabungkan teknik upsampling, SMOTE dan teknik down sampling Tomek links. Alasan pemilihan teknik ini sebab teknik ini bisa "men-generate" data baru pada kelas yang memiliki jumlah data sedikit dan sekaligus mengurangi kemungkinan terjadi overfitting.

Berikut adalah visualisasi persebaran nilai dalam "class"




Terlihat bahwa persebaran data sudah merata

## d. Feature Engineering

Berdasarkan *heatmap correlation* , kami menemukan hubungan terkuat terdapat antara fitur 2 dan fitur 3 , demikian juga antara fitur 3 dan fitur 5. Berdasarkan observasi tersebut, kami menguji banyak operasi matematika yang

menggabungkan pasangan tersebut menjadi satu kolom , seperti tabel yang ada di bawah ini

| Math Operation | F2 & F3 Importance | F2 & F5 Importance |
|---|---|---|
| Multiplication | 0.04020917 | 0.33217433 |
| Ratio | 0.02206169 | 0.48765212 |
| Difference | 0.03457015 | 0.05015583 |
| Sum | 0.16796541 | 0.07369298 |

Tabel diatas menunjukan bagaimana setiap kombinasi fitur berpengaruh terhadap model . Fitur yang lebih penting mengindikasikan bahwa fitur rekayasa yang baru memainkan peran besar dalam memprediksi variabel target . Oleh karena itu , kami membuat dua kolom baru, yang pertama adalah jumlah fitur 2 dan fitur 3 dan yang kedua adalah rasio antara fitur 2 dan fitur 3 .

Berdasarkan gambar.8 dan gambar.1 , kami menyimpulkan bahwa dengan membiarkan fitur track_name dan time_signature tidak diubah akan menyebabkan overfitting . jadi , kami memutuskan untuk membagi track_name (87%) dan lainnya (13%) . dan kami melakukan hal serupa dengan time_signature tipe 4 (91%) dan lainnya sebanyak (9%)

## e. Model Selection

Kita menggunakan 2 model untuk melakukan prediksi, yaitu XGBClassifier dan KNN.
Berikut adalah beberapa hasil prediksi score oleh masing masing model

| Parameter | Accuracy |
|---|---|
| {'random_state': 0, 'n_estimators': 200, 'max_depth': 8, 'learning_rate': 0.1} | 0.929861364 |
| {'random_state': 0, 'n_estimators': 300, 'max_depth': 6, 'learning_rate': 0.1} | 0.929770157 |
| {'random_state': 0, 'n_estimators': 100, 'max_depth': 9, 'learning_rate': 0.3} | 0.929496534 |
| {'random_state': 0, 'n_estimators': 100, 'max_depth': 6, 'learning_rate': 0.3} | 0.929496534 |
| {'random_state': 0, 'n_estimators': 200, 'max_depth': 9, 'learning_rate': 0.3} | 0.929405327 |

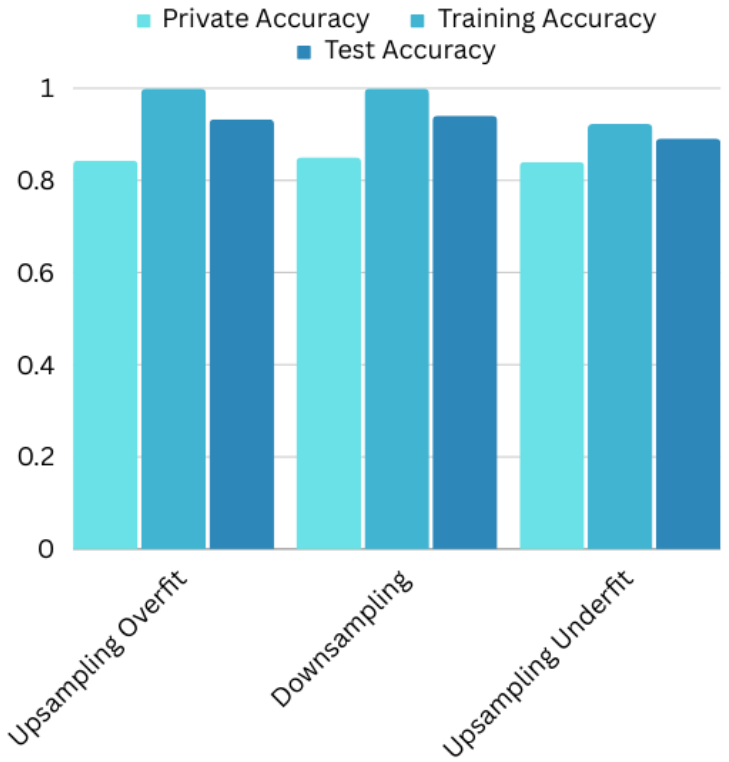
XGBClassifier

| Parameter | Accuracy |
|---|---|
| {'n_neighbors': 5} | 0.796280907 |
| {'n_neighbors': 15} | 0.763270055 |
| {'n_neighbors': 25} | 0.738830687 |
| {'n_neighbors': 35} | 0.722598256 |
| {'n_neighbors': 45} | 0.711929147 |

KNeighborsClassifier

## 3. Hasil dan Pembahasan/Analisis Hasil Prediksi

Setelah memilih model, kami hanya perlu membandingkan kombinasi preprocessing, fitur, dan parameters apa yang akan menghasilkan prediksi model terbaik.




Pada Bar Plot di atas, kami bisa menarik beberapa kesimpulan. Kesimpulan yang paling jelas adalah semua parameter relatif dekat satu sama lain , yaitu sekitar rentang akurasi 80% . downsampling memiliki private accuracy tertinggi dengan 0,85333 . pengamatan lebih lanjut menunjukkan bahwa unsampling underfit memiliki standar deviasi terkecil yaitu 0,03 menjadikannya model prediksi paling stabil . meskipun demikian , upsampling overfit berada di atas upsampling underfit dengan akurasi 0,8438 .

Oleh karena itu, kami memutuskan untuk menggunakan Upsampling Overfit untuk pengajuan akhir kami karena ini mewakili jalan tengah yang baik di antara kedua opsi tersebut, namun jika kami ingin meningkatkan model lebih jauh lagi, salah satu jalur yang mungkin adalah dengan melakukan underfit downsampling untuk mendapatkan model prediksi yang konsisten. dengan akurasi yang lebih tinggi .

# 4 Kesimpulan

Dari hasil analisa yang telah kami lakukan, kami menyimpulkan bahwa banyak metode yang bisa dipakai, tetapi XGBoost menghasilkan prediksi yang paling akurat. Dengan akurasi yang lebih dari 80% (berdasarkan scoring f1), kami dapat mendapatkan informasi yang bisa diandalkan untuk proses otomasi penyortiran genre musik. Tentu saja, ini bisa digunakan untuk memotong biaya tenaga kerja, karena mayoritas lagu sudah diberikan genre yang benar secara otomatis, beserta banyak manfaat lainnya.