Perbandingan Performa Metode Naive Bayes dan Support Vector Machine dalam Klasifikasi Genre Musik Berdasarkan Ekstraksi Fitur Sinyal Audio Menggunakan Mel-Frequency Cepstral Coefficients (MFCC)

by 09021282126083 Kevin Putrayudha Naserwan

Submission date: 17-Oct-2024 01:18PM (UTC+0700)
Submission ID: 2487951182
File name: si_Fitur_Sinyal_Audio_Menggunakan_Mel-Frequency_Cepstral_Co.docx (295.76K)
Word count: 13934
Character count: 91159

BAB I
PENDAHULUAN

1.1 Pendahuluan

Bab ini mencakup latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, serta sistematika penulisan skripsi yang berjudul "Perbandingan Performa Metode *Naive Bayes* dan *Support Vector Machine* dalam Klasifikasi *Genre* Musik Berdasarkan Ekstraksi Fitur Sinyal Audio Menggunakan *Mel-Frequency Cepstral Coefficients* (MFCC)."

1.2 Latar Belakang

Klasifikasi *genre* musik merupakan tantangan yang menarik di era digital, terutama dengan maraknya platform musik yang menggunakan teknologi untuk memberikan rekomendasi berdasarkan preferensi pengguna. Musik di setiap platform tersebut dikelompokkan ke dalam beberapa *genre* yang berbeda, sehingga pengguna dapat lebih mudah memilih jenis musik yang sesuai dengan preferensi mereka. Namun, dalam praktiknya, klasifikasi *genre* musik berdasarkan sinyal audio masih menjadi masalah yang kompleks dan membutuhkan pendekatan berbasis teknologi yang lebih akurat.

Dalam proses klasifikasi *genre* musik, ekstraksi fitur dari sinyal audio menjadi langkah krusial. Dua metode yang sering digunakan untuk tujuan ini adalah *Mel-Frequency Cepstral Coefficients* (MFCC) dan spektogram. Spektogram memberikan representasi visual dari spektrum frekuensi suara yang berubah seiring waktu, sementara MFCC mengonversi spektrum frekuensi ke skala mel yang lebih sesuai dengan persepsi pendengaran manusia (Müller, 2015).

Meskipun kedua metode ini memiliki kelebihan masing-masing, MFCC telah terbukti lebih efektif dalam banyak aplikasi pemrosesan sinyal audio,

termasuk klasifikasi *genre* musik. Penelitian yang dilakukan oleh Tzanetakis dan Cook (2002) menunjukkan bahwa MFCC memberikan performa yang lebih baik dibandingkan dengan fitur spektral lainnya dalam tugas klasifikasi *genre* musik. Selain itu, studi yang dilakukan oleh Lidy dan Rauber (2005) juga mengonfirmasi keunggulan MFCC dalam hal akurasi klasifikasi dan efisiensi komputasi dibandingkan dengan spektogram.

MFCC adalah metode yang sangat populer dalam pemrosesan sinyal audio karena kemampuannya untuk menangkap fitur akustik yang menyerupai persepsi manusia terhadap suara (Tzanetakis & Cook, 2002). Teknik ini umum digunakan dalam pengenalan ucapan dan musik, serta banyak diterapkan dalam sistem klasifikasi *genre* musik berbasis sinyal audio. Keunggulan MFCC dalam merepresentasikan karakteristik spektral suara yang relevan dengan persepsi manusia membuatnya menjadi pilihan yang tepat untuk penelitian ini.

Naive Bayes adalah salah satu algoritma pembelajaran mesin yang sering digunakan dalam klasifikasi. Algoritma ini didasarkan pada prinsip probabilitas dan bekerja dengan cukup baik dalam masalah klasifikasi yang sederhana. Salah satu keunggulan utama *Naive Bayes* adalah kesederhanaannya serta kemampuannya untuk diimplementasikan dengan cepat, meskipun algoritma ini membuat asumsi bahwa setiap fitur saling independen (Han et al., 2011). Dalam konteks klasifikasi *genre* musik, *Naive Bayes* dapat diterapkan dengan baik pada dataset yang besar dan bervariasi.

Di sisi lain, *Support Vector Machine* (SVM) adalah algoritma yang lebih kompleks dan sering digunakan dalam kasus klasifikasi yang melibatkan data berdimensi tinggi. SVM bekerja dengan menemukan *hyperplane* yang memisahkan kelas data dengan margin terbesar. Algoritma ini telah terbukti mampu memberikan hasil yang akurat dalam berbagai masalah klasifikasi, termasuk klasifikasi sinyal audio. SVM juga dapat menangani kasus di mana data tidak dapat dipisahkan secara linear dengan menggunakan teknik *kernel* (Hsu & Lin, 2002). Beberapa penelitian

menunjukkan bahwa SVM unggul dalam menangani masalah klasifikasi sinyal audio, terutama ketika data memiliki dimensi tinggi (Li et al., 2003).

Perbandingan antara *Naive Bayes* dan SVM dalam klasifikasi *genre* musik berdasarkan fitur MFCC menjadi penting untuk menentukan metode mana yang lebih baik dalam situasi tertentu. Masing-masing algoritma memiliki kelebihan dan kekurangan. *Naive Bayes*, dengan asumsi independensinya, bekerja dengan sangat cepat tetapi mungkin kurang akurat dalam masalah yang lebih kompleks. Sementara itu, SVM lebih kuat dalam hal akurasi, terutama ketika dihadapkan pada dataset yang lebih besar dan lebih kompleks, namun memerlukan waktu komputasi yang lebih lama.

Studi-studi terdahulu yang membandingkan metode-metode ini dalam berbagai aplikasi klasifikasi *genre* musik telah memberikan hasil yang bervariasi. Dalam beberapa kasus, *Naive Bayes* memberikan hasil yang cukup memuaskan ketika diterapkan pada dataset yang kecil dan kurang bervariasi. Namun, pada dataset yang lebih besar dan lebih kompleks, SVM cenderung memberikan akurasi yang lebih baik (Barchiesi et al., 2015). Oleh karena itu, perbandingan performa kedua metode ini dalam klasifikasi *genre* musik menjadi sangat penting untuk menentukan metode yang lebih optimal untuk diterapkan pada sistem klasifikasi musik secara otomatis.

1.3 Rumusan Masalah

Dari latar belakang yang telah dijelaskan, rumusan masalah penelitian ini berfokus pada perbandingan kinerja antara metode *Naive Bayes* dan *Support Vector Machine* pada klasifikasi *genre* musik. Penelitian ini mengeksplorasi seberapa efektif Mel-Frequency Cepstral Coefficients (MFCC) dapat diekstraksi dari sinyal audio untuk tujuan klasifikasi *genre* musik, serta menentukan metode mana yang menghasilkan klasifikasi paling akurat dalam kondisi MFCC yang digunakan. Penelitian ini juga mengkaji pengaruh MFCC terhadap kinerja klasifikasi *genre*

musik dengan kedua metode tersebut, serta menganalisis kelebihan dan kekurangan masing-masing metode berdasarkan karakteristik MFCC.

1.4 Tujuan Penelitian

Berdasarkan latar belakang dan rumusan masalah yang telah diuraikan, maka tujuan penelitian ini adalah sebagai berikut:

1. Memahami proses perbandingan kinerja metode *Naive Bayes* dan *Support Vector Machine* dalam klasifikasi berbagai sensor menggunakan ekstraksi fitur MFCC.
2. Memahami proses mengekstraksi fitur Koefisien Cepstral Frekuensi Mel (MFCC) dari sinyal audio untuk klasifikasi *genre* musik.
3. Untuk mengetahui keakuratan klasifikasi jenis *music* dengan metode *Naive Bayes* dan dukungan mesin vektor dengan kemampuan MFCC.
4. Menganalisis pengaruh parameter MFCC terhadap performa klasifikasi *genre* musik menggunakan kedua metode tersebut.
5. Mengidentifikasi kelebihan dan kekurangan masing-masing metode dalam konteks klasifikasi *genre* musik berdasarkan fitur MFCC.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah sebagai berikut:

1. Memberikan Pemahaman lebih dalam mengenai efektivitas metode *Naive Bayes* dan *Support Vector Machine* dalam mengklasifikasikan *genre* musik dapat membantu peneliti dan praktisi memilih metode yang sesuai untuk aplikasi serupa
2. Menyediakan informasi tentang keunggulan penggunaan fitur Mel-Frequency Cepstral Coefficients (MFCC) dalam ekstraksi karakteristik sinyal audio untuk klasifikasi *genre* musik, yang dapat dimanfaatkan dalam pengembangan sistem pengolahan audio lainnya.

3. Berkontribusi pada pengembangan sistem rekomendasi musik yang lebih akurat, yang dapat digunakan oleh platform streaming musik untuk meningkatkan pengalaman pengguna.
4. Membantu industri musik dalam mengotomatisasi proses kategorisasi dan pengarsipan musik berdasarkan *genre*, sehingga meningkatkan efisiensi manajemen konten musik.
5. Menyediakan dasar untuk penelitian lebih lanjut dalam bidang machine learning dan pengolahan sinyal audio, khususnya dalam konteks klasifikasi musik.
6. Memungkinkan pengembangan aplikasi musik yang lebih canggih, seperti sistem DJ otomatis atau alat bantu komposisi musik, yang dapat memanfaatkan hasil klasifikasi *genre* yang akurat.
7. Meningkatkan pemahaman tentang karakteristik akustik yang membedakan berbagai *genre* musik, yang dapat bermanfaat bagi studi musikologi dan analisis musik.

1.6 Batasan Masalah

Supaya penelitian ini tidak keluar dari topik diskusi, batasan yang akan dibahas adalah hal-hal berikut:

1. Perbandingan performa metode klasifikasi dibatasi pada *Naive Bayes* dan *Support Vector Machine* (SVM) untuk klasifikasi *genre* musik.
2. Ekstraksi fitur dari sinyal audio dibatasi pada penggunaan Mel-Frequency Cepstral Coefficients (MFCC) sebagai input untuk kedua metode klasifikasi.
3. Klasifikasi *genre* musik difokuskan pada *genre-genre* utama seperti pop, rock, jazz, klasik, dan elektronik, dengan jumlah *genre* yang terbatas untuk memastikan keseimbangan dataset.
4. Dataset yang digunakan terdiri dari rekaman audio musik dengan kualitas dan durasi yang seragam untuk memastikan konsistensi dalam proses ekstraksi fitur.

5. Evaluasi performa kedua metode akan diukur menggunakan metrik standar seperti akurasi, presisi, *recall*, dan *F1-score*.
6. Implementasi dan pengujian dilakukan menggunakan perangkat lunak dan libraries standar untuk machine learning dan pengolahan sinyal audio, tanpa optimisasi khusus untuk perangkat keras tertentu.

1.7 Sistematika Penulisan

Sistematika pembahasan dan penelitian dalam penulisan laporan tugas akhir ini terdiri dari beberapa bab sebagai berikut :

BAB I **PENDAHULUAN**

Bab ini menguraikan tentang latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, gaya penulisan, dan sistem penulisan

BAB II **TINJAUAN PUSTAKA**

Bab ini menjelaskan konsep alat dan komponen yang digunakan, serta perbandingan penelitian sebelumnya.

BAB III **METODOLOGI PENELITIAN**

Pada bab ini berisikan kerangka penelitian dan tahapan alur dari proses penelitian.

BAB IV **RANCANGAN PERANGKAT LUNAK**

Pada bab ini menjelaskan arsitektur sistem dan alur proses dari penelitian.

BAB V **HASIL DAN PEMBAHASAN**

Bab ini memaparkan hasil penelitian dan membahas faktor-faktor yang memengaruhi hasil.

BAB VI **KESIMPULAN DAN SARAN**

Pada Bab ini menyimpulkan hasil penelitian terkait dan memberikan saran untuk pengembangan lebih lanjut..

1.8 Kesimpulan

Pada bab ini telah dijelaskan latar belakang masalah dilakukannya perbandingan performa metode *Naive Bayes* dan *Support Vector Machine* dalam klasifikasi *genre* musik berdasarkan ekstraksi fitur sinyal audio menggunakan Mel-Frequency Cepstral Coefficients (MFCC). Bab ini telah menguraikan pentingnya klasifikasi *genre* musik di era digital, tantangan yang dihadapi dalam proses klasifikasi, serta potensi penggunaan metode machine learning dan ekstraksi fitur MFCC untuk mengatasi tantangan tersebut.

Dari pembahasan tersebut, telah dirumuskan masalah penelitian yang berfokus pada perbandingan efektivitas *Naive Bayes* dan SVM dalam klasifikasi *genre* musik menggunakan fitur MFCC. Tujuan penelitian telah ditetapkan untuk mengetahui proses perbandingan kedua metode, memahami proses kerja ekstraksi fitur MFCC, menganalisis tingkat akurasi klasifikasi, serta mengidentifikasi kelebihan dan kekurangan masing-masing metode.

Bab ini juga telah menguraikan manfaat penelitian yang mencakup kontribusi pada pengembangan sistem rekomendasi musik, otomatisasi kategorisasi musik, dan peningkatan pemahaman tentang karakteristik akustik berbagai *genre* musik. Batasan masalah telah ditetapkan untuk memfokuskan penelitian pada perbandingan *Naive Bayes* dengan SVM menggunakan MFCC sebagai metode untuk ekstraksi fitur.

Akhirnya, sistematika penulisan telah diuraikan untuk memberikan gambaran struktur keseluruhan skripsi. Dengan demikian, bab ini telah memberikan landasan yang kuat untuk pelaksanaan penelitian dan pembahasan lebih lanjut dalam bab-bab berikutnya.

BAB II
TINJAUAN PUSTAKA

2.1 Pendahuluan

Bab ini menyajikan landasan teori yang mendukung penelitian tentang klasifikasi *genre* musik menggunakan metode machine learning. Dimulai dengan penjelasan umum tentang machine learning, bab ini kemudian membahas konsep klasifikasi *genre* musik. Selanjutnya, diuraikan secara rinci tentang Mel-Frequency Cepstral Coefficients (MFCC), termasuk tahapan-tahapannya seperti *pre-emphasis*, *framing* dan *windowing*, *Fourier Transform*, *filterbank Mel Scale*, DCT, serta penerapannya dalam klasifikasi *genre* musik. Bab ini juga menjelaskan dua metode klasifikasi utama: *Naive Bayes* dan *Support Vector Machine* (SVM), termasuk prinsip kerja, persamaan, fungsi *kernel*, penerapan, kelebihan, dan keterbatasan SVM. Terakhir, disajikan tinjauan penelitian terdahulu yang relevan dengan topik ini.

2.2 Machine Learning

Pembelajaran mesin adalah metode pembuatan program yang dapat belajar dari data, tidak seperti program statis tradisional Program pembelajaran mesin dirancang untuk belajar secara mandiri dengan menganalisis pola dari data yang diberikan, seperti halnya manusia belajar dari contoh Proses ini memungkinkan program menemukan pola dalam "kotak hitam" yang mewakili fungsi matematika yang tidak diketahui dengan mengamati input dan output yang ada, program berupaya mengembangkan formula atau model yang hampir mewakili input dan output tersebut (Hiadayat et al., 2020).

Sebagai bidang studi, machine learning berfokus pada pengembangan algoritma yang memungkinkan komputer belajar dari data tanpa perlu diprogram secara eksplisit. Seperti yang diungkapkan oleh Samuel, algoritma ini bersifat umum dan dapat menghasilkan aturan atau model berdasarkan data yang tersedia,

tanpa perlu kode khusus. Contohnya, algoritma pengenalan tulisan tangan dapat digunakan untuk membedakan email spam dan non-spam tanpa perubahan dalam kodenya, karena ia menghasilkan logika klasifikasi berdasarkan data yang berbeda.

Selain itu, pembelajaran mesin dapat digambarkan sebagai kemampuan komputer untuk belajar dari data dan membuat keputusan berdasarkan model yang dibangun dari data tersebut (I. Daqiqil, 2021). ada tiga kategori utama pembelajaran mesin: pembelajaran yang diawasi, pembelajaran tanpa pengawasan, dan pembelajaran penguatan (Homepage et al., 2019).

Supervised Learning menggunakan data yang telah diberi label untuk melatih model, dan umumnya diterapkan dalam dua kategori utama: klasifikasi dan regresi. Klasifikasi melibatkan pemetaan output ke kategori, seperti menentukan warna merah atau biru, sementara regresi berfokus pada menghasilkan nilai riil, seperti jumlah uang atau berat badan. Berbagai algoritma digunakan dalam supervised learning, seperti Backpropagation, Random Forest, dan *Support Vector Machines* (SVM). Algoritma populer untuk klasifikasi termasuk SVM, *K-Nearest Neighbor* (KNN), dan *Artificial Neural Networks* (ANN).

Di sisi lain, *Unsupervised* Learning beroperasi tanpa label data, dengan tujuan mengidentifikasi pola tersembunyi dalam data. Metode ini terutama terdiri dari pengelompokan (*clustering*) dan asosiasi. *Clustering*, seperti *k-means*, mengelompokkan entitas berdasarkan kesamaan, sementara asosiasi, seperti algoritma Apriori, menemukan hubungan antara item dalam data, misalnya kebiasaan pembelian.

Sementara itu, *Reinforcement Learning* menggunakan pendekatan berdasarkan proses pengambilan keputusan Markov (*Markov Decision Process*). Dua pendekatan utama dalam metode ini adalah metode berbasis model, seperti algoritma SARSA, yang mempelajari proses sebelum mengembangkan strategi optimal, dan metode tanpa model, seperti *Q-learning*, yang menghitung strategi

optimal langsung dari interaksi tanpa mengetahui model proses sebelumnya (Homepage et al., 2019).

2.3 Klasifikasi *Genre* Musik

Klasifikasi *genre* musik merupakan salah satu tugas penting dalam bidang *Music* Information Retrieval (MIR) yang semakin relevan seiring dengan pesatnya perkembangan platform musik digital seperti Spotify, Apple *Music*, dan Pandora (Dieleman & Schrauwen, 2014; Schedl et al., 2014). Sistem-sistem ini menggunakan algoritma canggih untuk menganalisis dan mengkategorikan musik ke dalam *genre* tertentu, yang kemudian digunakan untuk merekomendasikan lagu kepada pengguna berdasarkan preferensi mereka (Dieleman & Schrauwen, 2014).

Proses klasifikasi *genre* musik melibatkan pengelompokan sinyal audio ke dalam kategori *genre* yang telah ditentukan sebelumnya dengan menggunakan berbagai teknik pembelajaran mesin. Teknik-teknik ini dirancang untuk mengidentifikasi dan menganalisis berbagai elemen musikal yang membedakan satu *genre* dari yang lain, termasuk :

1. Tempo: Kecepatan atau pace dari musik, yang sering kali menjadi ciri khas *genre* tertentu. Misalnya, musik dance elektronik umumnya memiliki tempo yang lebih cepat dibandingkan dengan balada (Gouyon & Dixon, 2005).
2. Harmoni: Kombinasi nada yang dimainkan secara bersamaan, yang dapat sangat bervariasi antar *genre*. Jazz, misalnya, sering menggunakan struktur harmoni yang lebih kompleks dibandingkan dengan musik pop (Ewert, 2011).
3. Timbre: Karakteristik suara yang membedakan satu instrumen atau suara dari yang lain, bahkan ketika mereka memainkan nada yang sama. Timbre sangat penting dalam membedakan *genre* seperti rock (yang sering didominasi oleh gitar listrik) dari klasik (yang lebih banyak menggunakan instrumen orkestra) (Peeters et al., 2011).

4. Ritme: Pola ketukan dalam musik, yang dapat sangat berbeda antar *genre*. Misalnya, ritme syncopated dalam jazz berbeda signifikan dengan beat yang lebih teratur dalam banyak lagu pop (Sturm, 2014).
5. Instrumentasi: Jenis dan kombinasi instrumen yang digunakan, yang sering kali menjadi penanda kuat dari *genre* tertentu. Misalnya, penggunaan synthesizer yang menonjol dalam musik elektronik, atau dominasi gitar akustik dalam folk *music* (Heittola et al., 2009).
6. Struktur lagu: Bagaimana berbagai bagian lagu (seperti verse, chorus, bridge) disusun, yang dapat bervariasi secara signifikan antar *genre* (Paulus et al., 2010).

Untuk mengekstrak dan menganalisis fitur-fitur ini dari sinyal audio, peneliti dan pengembang menggunakan berbagai teknik pemrosesan sinyal digital. Salah satu metode yang paling umum digunakan adalah *Mel-Frequency Cepstral Coefficients* (MFCC), yang efektif dalam menangkap karakteristik spektral dari sinyal audio (Logan, 2000). Teknik lain termasuk analisis chroma, yang berfokus pada distribusi energi pitch dalam musik (Ewert, 2011), dan spectral flux, yang mengukur bagaimana spektrum frekuensi berubah dari waktu ke waktu (Tzanetakis & Cook, 2002).

Setelah fitur-fitur ini diekstrak, algoritma pembelajaran mesin seperti *Support Vector Machine*s (SVM), *Naive Bayes*, dan lebih baru-baru ini, Deep Neural Networks (DNN), digunakan untuk mempelajari pola-pola yang membedakan satu *genre* dari yang lain (Choi et al., 2017). Pendekatan deep learning, seperti Convolutional Neural Networks (CNN) dan Recurrent Neural Networks (RNN), telah berhasil dalam beberapa tahun terakhir dengan mampu menangkap fitur-fitur kompleks dan dependensi temporal dalam musik (Pons et al., 2017).

Meskipun telah ada kemajuan signifikan dalam bidang ini, klasifikasi *genre* musik tetap menjadi tantangan yang kompleks. Ini disebabkan oleh sifat subjektif dari definisi *genre*, overlap antar *genre*, dan evolusi *genre* musik dari waktu ke

waktu (Sturm, 2014). Selain itu, banyak lagu modern yang menggabungkan elemen dari berbagai *genre*, yang semakin mempersulit tugas klasifikasi (Pachet & Cazaly, 2000).

Terlepas dari tantangan-tantangan ini, penelitian dalam klasifikasi *genre* musik terus berkembang, didorong oleh aplikasi praktisnya dalam industri musik digital dan potensinya untuk meningkatkan pengalaman pendengar musik (Knees & Schedl, 2014). Kemajuan dalam bidang ini tidak hanya bermanfaat untuk sistem rekomendasi musik, tetapi juga memiliki implikasi yang lebih luas dalam pemahaman kita tentang struktur dan karakteristik musik, serta bagaimana kita mengkategorikan dan memahami seni suara ini (Herremans et al., 2017).

2.4 *Mel-Frequency Cepstral Coefficients* (MFCC)

MFCC adalah teknik yang populer dalam pengolahan sinyal audio, terutama dalam pengenalan ucapan dan musik. MFCC mampu merepresentasikan spektrum frekuensi suara yang berhubungan erat dengan persepsi suara manusia. Dalam konteks klasifikasi *genre* musik, MFCC digunakan sebagai fitur untuk menangkap pola dari sinyal audio yang dapat membedakan antara *genre* yang berbeda (Costa et al., 2017).

MFCC bekerja dengan mengonversi sinyal suara mentah ke dalam bentuk yang lebih representatif dengan menguraikan sinyal menjadi beberapa koefisien. Koefisien ini digunakan oleh algoritma pembelajaran mesin untuk menganalisis dan mengklasifikasikan sinyal audio berdasarkan *genre*.

2.4.1 *Pre-emphasis*

Pada tahap awal, sinyal audio biasanya melewati filter *pre-emphasis*, yang bertujuan untuk menekankan komponen frekuensi tinggi pada sinyal suara. Ini dilakukan karena sinyal frekuensi tinggi biasanya memiliki energi yang lebih

rendah dibandingkan frekuensi rendah, sehingga dapat memperkuat sinyal dan membuat analisis lebih efektif (Davis, 1980).

- **Persamaan *Pre-emphasis*:**

$$y[n]=x[n]-\alpha \cdot x[n-1] \quad \text{(II-1)}$$

Di mana x[n]x[n] adalah sinyal input, y[n]y[n] adalah sinyal output, dan αα adalah konstanta *pre-emphasis* (biasanya bernilai 0,97).

2.4.2 *Framing* dan *Windowing*

Sinyal suara bersifat dinamis, berubah dari waktu ke waktu. Oleh karena itu, sinyal suara harus dipotong menjadi frame-frame kecil (biasanya sekitar 20-40 ms). Tiap frame dianggap stasioner selama waktu singkat ini, dan analisis dilakukan untuk tiap frame (Logan, 2000).

Setelah *framing*, *windowing* diterapkan untuk mengurangi efek "discontinuities" pada batas frame. *Windowing* yang paling umum digunakan adalah *Hamming Window*, yang diberikan oleh:

$$w(n) = 0.54 - 0.46 \cdot \cos\left(\frac{2\pi n}{N-1}\right) \quad \text{(II-2)}$$

Penjelasan tiap komponen dari rumus tersebut:

- w(n): Fungsi window Hamming yang diterapkan pada sampel ke-n.
- n: Indeks dari sampel dalam frame, dengan nilai $0 \leq n \leq N-1$.
- N: Panjang dari window atau jumlah total sampel dalam satu frame.
- Cos $\left(\frac{2\pi n}{N-1}\right)$: Fungsi cosinus yang memperhalus transisi di tepi window, sehingga dapat mengurangi diskontinuitas antara frame.
- 0.54 dan 0.46: Koefisien yang menentukan bentuk karakteristik dari Hamming window.

Windowing digunakan untuk mengurangi efek diskontinuitas pada batas frame. Hamming window memberikan bobot yang lebih besar pada sampel-sampel di tengah frame dan bobot yang lebih kecil pada sampel-sampel di tepi frame, yang membantu menghaluskan transisi antara frame satu dan lainnya, sehingga analisis frekuensi sinyal suara menjadi lebih akurat.

2.4.3 *Fourier Transform* dan Spektrogram

Transformasi Fourier mentransformasikan sinyal dari domain waktu ke domain frekuensi, menghasilkan spektogram yang menampilkan frekuensi sinyal terhadap waktu. Hal ini membantu dalam mengekstraksi karakteristik frekuensi sinyal audio (O'Shaughnessy, 2000).

2.4.4 *Filterbank Mel Scale*

Mel Scale adalah skala yang lebih sesuai dengan persepsi pendengaran manusia. Untuk meniru cara telinga manusia merespons suara, spektrum frekuensi diubah ke skala Mel menggunakan filter bank segitiga yang berfokus pada frekuensi yang lebih rendah (O'Shaughnessy, 2000).

Formula konversi frekuensi dari Hertz ke Mel diberikan oleh:

$$M(f) = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \quad \text{(II-3)}$$

Penjelasan tiap komponen dari rumus tersebut:

- $M(f)$: Frekuensi dalam skala Mel, yang merepresentasikan bagaimana manusia merasakan frekuensi suara.
- f: Frekuensi dalam Hertz, yang merupakan unit frekuensi standar yang biasanya digunakan dalam pengukuran fisik.
- 2595: Faktor yang digunakan untuk mengalihkan skala ke satuan Mel, hasil dari penelitian tentang persepsi suara manusia.
- log 10: Logaritma basis 10, yang digunakan untuk menciptakan hubungan non-linear antara skala Hertz dan skala Mel, agar lebih sesuai dengan cara telinga manusia menangkap frekuensi suara.
- $\left(1 + \frac{f}{700}\right)$: Bagian dari formula yang menggeser dan menormalkan nilai frekuensi ff dalam Hertz, di mana 700 adalah konstanta yang mendefinisikan titik referensi untuk transisi antara skala linier di frekuensi rendah dan skala logaritmik di frekuensi yang lebih tinggi.

Fungsi konversi ini digunakan dalam banyak aplikasi pengolahan sinyal suara, khususnya dalam ekstraksi fitur untuk pengenalan suara, karena lebih mencerminkan bagaimana manusia mendengar perubahan frekuensi pada berbagai rentang. Skala Mel memiliki resolusi yang lebih tinggi pada frekuensi rendah dan resolusi yang lebih rendah pada frekuensi tinggi, mirip dengan persepsi pendengaran manusia.

2.4.5 DCT (*Discrete Cosine Transform*)

Setelah melalui filterbank Mel, logaritma amplitudo spektrum diambil, kemudian diterapkan *Discrete Cosine Transform* (DCT). DCT mengkompres informasi dengan mengubahnya ke dalam domain Cepstral, di mana koefisien frekuensi tertinggi dibuang. Hasilnya adalah koefisien MFCC, yang biasanya terdiri dari 12 hingga 13 nilai per frame (Tzanetakis & Cook, 2002).

2.4.6 Delta dan Delta-Delta MFCC

Untuk menangkap dinamika perubahan sinyal audio dari waktu ke waktu, koefisien MFCC diperpanjang dengan komponen delta dan delta-delta. Fitur ini mencakup informasi tentang kecepatan dan percepatan perubahan spektrum audio, yang sangat berguna dalam klasifikasi musik (Picone, 1993).

2.4.7 Penerapan MFCC dalam Klasifikasi *Genre* Musik

Dalam klasifikasi *genre* musik, MFCC digunakan sebagai fitur yang merepresentasikan sinyal audio berdasarkan pola frekuensi yang ada di setiap *genre*. Misalnya, *genre* musik rock, jazz, dan klasik memiliki karakteristik spektrum frekuensi yang berbeda, dan MFCC dapat menangkap perbedaan ini untuk tujuan klasifikasi. Fitur MFCC dapat digunakan bersama algoritma pembelajaran mesin seperti *Naive Bayes* dan *Support Vector Machine* (SVM) untuk melakukan klasifikasi *genre* musik secara efektif (Li et al., 2003).

Diagram Proses MFCC :

2.5 *Naive Bayes*

Naive Bayes adalah metode klasifikasi probabilistik yang didasarkan pada teorema Bayes. Metode ini disebut "naif" karena mengasumsikan bahwa semua fitur dalam dataset bersifat independen satu sama lain, meskipun dalam kenyataannya asumsi ini jarang terpenuhi sepenuhnya. Meski demikian, *Naive Bayes* tetap menjadi metode yang populer dan efektif untuk berbagai tugas klasifikasi dalam machine learning (Zhang, 2004; Rish, 2001).

Naive Bayes menggunakan probabilitas bersyarat untuk memprediksi kelas dari instance baru berdasarkan nilai-nilai fiturnya. Metode ini menghitung probabilitas suatu instance termasuk dalam kelas tertentu berdasarkan fitur-fitur

yang diekstraksi dari data. Teorema Bayes yang menjadi dasar dari metode ini dapat dinyatakan sebagai berikut:

$$P(C|X) = P(X|C) * P(C) / P(X) \quad \text{(II-4)}$$

Di mana:

- P(C|X) merupakan probabilitas posterior, yaitu probabilitas *Class* C given fitur X
- P(X|C) merupakan *likelihood*, yaitu probabilitas fitur X given *Class* C
- P(C) merupakan probabilitas prior dari *Class* C
- P(X) merupakan *evidence*, yaitu probabilitas fitur X

Dalam konteks *Naive Bayes*, kita mengasumsikan independensi fitur, sehingga:

$$P(X|C) = P(x1|C) * P(x2|C) * ... * P(xn|C) \quad \text{(II-5)}$$

Di mana x1, x2, ..., xn adalah fitur-fitur individu.

Untuk klasifikasi, kita mencari kelas dengan probabilitas posterior tertinggi:

$$C^* = \text{argmax}[C]\ P(C|X) = \text{argmax}[C]\ P(X|C) * P(C) / P(X) \quad \text{(II-6)}$$

Karena P(X) konstan untuk semua kelas, kita bisa mengabaikannya:

$$C^* = \text{argmax}[C]\ P(X|C) * P(C) \quad \text{(II-7)}$$

Dalam konteks klasifikasi menggunakan *Naive Bayes*, langkah-langkahnya umumnya adalah sebagai berikut:

1. Ekstraksi Fitur: Data diproses untuk mengekstrak fitur-fitur yang relevan.
2. Training: Model *Naive Bayes* dilatih menggunakan dataset dengan label kelas yang sudah diketahui.

3. Klasifikasi: Untuk data baru, fitur-fiturnya diekstrak dan dimasukkan ke dalam model *Naive Bayes* untuk memprediksi kelasnya.

Kelebihan *Naive Bayes* dalam klasifikasi antara lain:

1. Efisiensi komputasi: Metode ini cepat dalam training dan prediksi.
2. Performa baik dengan dataset kecil: *Naive Bayes* dapat memberikan hasil yang baik bahkan dengan jumlah data training yang terbatas.
3. Mudah diimplementasikan: Kesederhanaan algoritma membuatnya mudah untuk diterapkan dan dimodifikasi.
4. Skalabilitas: Dapat menangani sejumlah besar fitur dengan baik.

Namun, perlu diingat bahwa asumsi independensi fitur dalam *Naive Bayes* mungkin tidak selalu terpenuhi dalam data nyata, yang bisa mempengaruhi akurasi dalam beberapa kasus. Beberapa penelitian telah membandingkan *Naive Bayes* dengan metode lain seperti SVM dan neural networks, menunjukkan bahwa meskipun *Naive Bayes* memiliki kelebihan dalam kesederhanaan dan kecepatan, metode lain mungkin memberikan akurasi yang lebih tinggi dalam beberapa skenario (Zhang, 2004; Rish, 2001).

2.6 *Support Vector Machine* (SVM)

Support Vector Machine (SVM) merupakan salah satu algoritma pembelajaran mesin yang paling kuat dan cerdas yang digunakan untuk tugas klasifikasi dan denoising. Metode SVM bekerja dengan mencari database optimal yang dapat memisahkan kelas-kelas yang berbeda dalam ruang multi-fitur. Tujuannya adalah untuk memaksimalkan margin antara kelas-kelas. (Kurts dan Vapnik, 1995).

2.6.1 Prinsip Kerja

1. Transformasi Data: SVM dapat menangani kasus klasifikasi yang tidak dapat dipisahkan secara linier dengan menggunakan teknik *kernel trick*. *Kernel trick* ini mentransformasikan data ke dalam ruang dimensi yang lebih tinggi, memungkinkan data yang awalnya tidak terpisah secara linier menjadi dapat dipisahkan oleh *hyperplane* (Cortes & Vapnik, 1995; Boser et al., 1992).
2. Maximizing Margin: SVM berusaha menemukan *hyperplane* yang memaksimalkan margin antara kelas yang berbeda. Margin adalah jarak terpendek antara *hyperplane* dan titik data dari setiap kelas. Dengan margin yang lebih besar, SVM meningkatkan generalisasi dan mengurangi risiko kesalahan klasifikasi pada data yang belum terlihat (Bennett & Campbell, 2000).
3. Support Vectors: Titik-titik data terdekat dengan *hyperplane*, yang disebut sebagai *support vectors*, sangat menentukan posisi optimal dari *hyperplane*. Hanya *support vectors* ini yang mempengaruhi *hyperplane*, dan bukan titik data lainnya (Vapnik, 1998).

2.6.2 Persamaan SVM

Untuk kasus linier, fungsi SVM diuraikan sebagai:

$$f(x) = sign(w^t x + b) \tag{II-8}$$

di mana:

- w adalah vektor bobot tegak lurus terhadap *hyperplane*,
- x adalah vektor input (fitur),
- b adalah bias yang menggeser *hyperplane*.

SVM berusaha untuk memaksimalkan margin, yang dapat diformulasikan sebagai masalah optimasi:

$$\frac{min}{w,b} \frac{1}{2} ||w||^2 \tag{II-9}$$

dengan kendala:

$$y_i(w^t x_i + b) \geq 1 \quad \forall i \tag{II-10}$$

di mana y_i merupakan label kelas (1 atau -1) dari titik data x_i.

2.6.3 Fungsi *Kernel*

Untuk kasus non-linier, SVM memakai fungsi *kernel* untuk mentransformasikan data ke ruang dimensi yang lebih tinggi tanpa perlu secara eksplisit melakukan transformasi tersebut. Salah satu *kernel* yang umum digunakan adalah *kernel* Radial Basis Function (RBF):

$$K(x, x') = exp(-\gamma||x - x'||^2) \tag{II-11}$$

di mana:

- $K(x, x')$ merupakan fungsi *kernel* yang mengukur kesamaan antara dua titik data x dan x'
- γ adalah parameter *kernel* yang mengontrol lebar dari fungsi Gaussian.

2.6.4 Penerapan dalam Klasifikasi

1. Ekstraksi Fitur: SVM bekerja pada dataset yang sudah dikonversi ke dalam bentuk fitur, di mana setiap fitur mewakili karakteristik penting dari data.
2. Training: Model SVM dilatih dengan menggunakan data yang berlabel, di mana SVM akan mencoba menemukan *hyperplane* optimal yang memisahkan kelas-kelas berdasarkan fitur-fitur tersebut.
3. Klasifikasi: Setelah dilatih, model SVM bisa digunakan untuk mengklasifikasikan data baru dengan memprediksi label berdasarkan posisi relatif data terhadap *hyperplane*.

2.6.5 Kelebihan SVM

1. Performa tinggi: SVM memiliki akurasi yang tinggi, terutama pada data dengan batas keputusan yang kompleks atau non-linier (Cortes & Vapnik, 1995).
2. Efektif dalam ruang dimensi tinggi: SVM sangat cocok untuk dataset dengan banyak fitur, karena algoritma ini tidak terpengaruh oleh *curse of dimensionality* (Vapnik, 1998).
3. Fleksibel: Dengan berbagai pilihan *kernel*, SVM dapat digunakan untuk menangani berbagai jenis data, baik yang linier maupun non-linier (Boser et al., 1992).

2.6.6 Keterbatasan

1. Kompleksitas komputasi: SVM bisa menjadi lambat untuk dataset yang sangat besar, terutama saat harus melakukan *training* pada data non-linier (Bennett & Campbell, 2000).
2. Pemilihan *kernel* dan parameter: SVM memerlukan pemilihan *kernel* dan tuning parameter yang cermat untuk mencapai performa optimal. Penggunaan *cross-validation* sering kali diperlukan untuk menentukan parameter terbaik (Hsu et al., 2003).

2.7 Penelitian Terdahulu

Penelitian-penelitian terdahulu mengenai klasifikasi *genre* musik telah banyak memberikan kontribusi dalam memahami efektivitas metode ekstraksi fitur dan algoritma pembelajaran mesin. Beberapa penelitian penting yang berkaitan dengan penggunaan MFCC dan algoritma pembelajaran mesin untuk klasifikasi *genre* musik dijelaskan di bawah ini.

2.7.1 Tzanetakis & Cook (2002)

Penelitian yang dilakukan oleh Tzanetakis dan Cook pada tahun 2002 menjadi salah satu studi awal yang menyoroti penggunaan MFCC dalam klasifikasi *genre* musik. Mereka mengembangkan sistem klasifikasi musik otomatis yang menggunakan MFCC sebagai fitur utama dan membandingkannya dengan berbagai fitur spektral lainnya. Hasil penelitian mereka menunjukkan bahwa MFCC memiliki keunggulan dalam hal akurasi dan efisiensi komputasi untuk tugas klasifikasi musik, membuatnya populer dalam pemrosesan sinyal audio.

2.7.2 Lidy & Rauber (2005)

Lidy dan Rauber melanjutkan penelitian tentang penggunaan MFCC dalam klasifikasi musik. Mereka melakukan studi perbandingan antara MFCC dan metode lainnya, seperti spektogram, untuk menilai efektivitas ekstraksi fitur pada sinyal audio. Hasil penelitian menunjukkan bahwa MFCC memberikan akurasi yang lebih baik dalam sebagian besar kasus. Penelitian mereka semakin memperkuat posisi MFCC sebagai metode yang efektif untuk representasi akustik dalam klasifikasi *genre* musik.

2.7.3 Li et al. (2003)

Li et al. meneliti pentingnya pemilihan algoritma klasifikasi yang tepat untuk meningkatkan akurasi dalam tugas klasifikasi musik. Penelitian mereka membandingkan beberapa algoritma pembelajaran mesin, termasuk *Support Vector Machine* (SVM), dalam konteks klasifikasi musik. Mereka menemukan bahwa SVM unggul dalam menangani dataset berdimensi tinggi dan memberikan akurasi yang lebih baik dibandingkan algoritma lain ketika digunakan bersama dengan MFCC sebagai fitur input.

2.7.4 Hsu & Lin (2002)

Hsu dan Lin fokus pada pengembangan dan penerapan SVM dalam berbagai masalah klasifikasi, termasuk klasifikasi sinyal audio. Penelitian mereka menunjukkan bagaimana SVM dapat memisahkan data dengan margin terbesar,

yang menghasilkan performa yang sangat baik dalam klasifikasi musik, terutama ketika menggunakan teknik *kernel* untuk mengelola data yang tidak dapat dipisahkan secara linear. SVM menjadi salah satu dari algoritma yang sangat sering dipakai dalam klasifikasi musik berdasarkan sinyal audio.

2.7.5 Barchiesi et al. (2015)

Penelitian Barchiesi et al. menyoroti perbandingan antara *Naive Bayes* dan SVM dalam klasifikasi *genre* musik pada dataset besar dan kompleks. Mereka menemukan bahwa meskipun *Naive Bayes* bekerja dengan cepat dan efisien pada dataset yang lebih kecil, SVM menunjukkan keunggulan dalam akurasi pada dataset yang lebih besar. Penelitian ini memberikan wawasan penting mengenai kapan setiap algoritma dapat lebih optimal dalam tugas klasifikasi.

2.7.6 Costa et al. (2017)

Penelitian oleh Costa et al. menunjukkan penerapan Convolutional Neural Networks (CNN) dalam klasifikasi *genre* musik sebagai alternatif dari algoritma tradisional seperti SVM. Mereka menemukan bahwa CNN mampu menghasilkan akurasi yang sangat baik dalam klasifikasi *genre* musik, namun memerlukan lebih banyak sumber daya komputasi. Studi ini memberikan perspektif baru tentang bagaimana metode deep learning dapat bersaing dengan algoritma klasik seperti SVM dan *Naive Bayes*.

2.8 Kesimpulan

Bab ini telah memberikan landasan teoritis yang komprehensif untuk penelitian tentang klasifikasi *genre* musik menggunakan metode machine learning. Pembahasan dimulai dengan penjelasan tentang machine learning dan aplikasinya dalam pengolahan sinyal audio, khususnya dalam konteks klasifikasi *genre* musik. Teknik ekstraksi fitur Mel-Frequency Cepstral Coefficients (MFCC) telah diuraikan secara rinci, termasuk tahapan-tahapan proses seperti *pre-emphasis*,

framing dan *windowing*, *Fourier Transform*, *filterbank Mel Scale*, dan *Discrete Cosine Transform* (DCT). Penjelasan ini memberikan pemahaman mendalam tentang bagaimana MFCC dapat menangkap karakteristik spektral sinyal audio yang relevan dengan persepsi pendengaran manusia.

Naive Bayes serta *Support Vector Machine* (SVM), telah dibahas secara menyeluruh. Prinsip kerja, formulasi matematis, kelebihan, dan keterbatasan masing-masing metode telah diuraikan, memberikan dasar yang kuat untuk pemahaman dan implementasi kedua algoritma ini dalam konteks klasifikasi *genre* musik.

Tinjauan literatur yang disajikan menunjukkan perkembangan penelitian dalam bidang klasifikasi *genre* musik, menekankan efektivitas MFCC sebagai metode ekstraksi fitur dan potensi algoritma pembelajaran mesin seperti SVM dalam meningkatkan akurasi klasifikasi.

Secara keseluruhan, bab ini telah menyediakan kerangka teoritis yang diperlukan untuk memahami kompleksitas klasifikasi *genre* musik dan memberikan justifikasi untuk penggunaan MFCC, *Naive Bayes*, dan SVM dalam penelitian ini. Pemahaman ini akan menjadi dasar untuk metodologi penelitian, implementasi, dan analisis hasil yang akan dijelaskan dalam bab-bab selanjutnya.

BAB III
METODOLOGI PENELITIAN

BAB IV
RANCANGAN PERANGKAT LUNAK

4.1 Pendahuluan

Bab ini menjelaskan secara rinci proses pengembangan perangkat lunak untuk klasifikasi *genre* musik menggunakan algoritma *Naive Bayes* dan *Support Vector Machine* (SVM). Pengembangan perangkat lunak ini mengikuti Metode Pengembangan Sistem Waterfall, yang terdiri dari lima tahapan utama: Analisis, Desain, Implementasi (Coding), Pengujian, dan Pemeliharaan. Melalui pendekatan ini, bertujuan untuk menciptakan sistem yang *robust* dan efisien dalam mengklasifikasikan *genre* musik berdasarkan fitur yang diekstraksi dari sinyal audio menggunakan *Mel-Frequency Cepstral Coefficients* (MFCC).

4.2 Fase Analisis

Pada fase analisis, dilakukan pemetaan kebutuhan bisnis dan teknis yang berhubungan terhadap system yang ingin dibangun, serta identifikasi kebutuhan perangkat lunak untuk mendukung pengembangan aplikasi klasifikasi *genre* musik.

4.2.1 Pemodelan Bisnis

Pemodelan bisnis di perangkat lunak ini melibatkan penentuan skenario penggunaan sistem, di mana perangkat lunak ini diharapkan dapat digunakan oleh pengguna yang ingin mengklasifikasikan *genre* musik secara otomatis berdasarkan file audio yang mereka miliki. Model bisnis berfokus pada efisiensi dan akurasi dalam pemrosesan data, yang memungkinkan penggunaan metode machine learning seperti *Naive Bayes* dan SVM untuk meningkatkan performa klasifikasi.

4.2.2 Kebutuhan Sistem

Untuk klasifikasi jenis musik, hasil klasifikasi harus benar, dan persyaratan komputer yang diperlukan dalam klasifikasi musik, berdasarkan pemodelan komersial, fitur-fitur yang disediakan oleh komputer seperti:

1. Upload File Audio

   Pengguna dapat mengunggah file audio untuk diproses oleh sistem.
2. Ekstraksi Fitur Audio

   Sistem secara otomatis mengekstraksi fitur audio menggunakan metode *Mel-Frequency Cepstral Coefficients* (MFCC), yang merupakan representasi numerik dari karakteristik frekuensi audio.
3. Pemilihan Model Klasifikasi

   Pengguna dapat memilih untuk melakukan klasifikasi menggunakan algoritma *Naive Bayes* atau SVM.
4. Visualisasi Hasil

   Perangkat lunak menampilkan hasil klasifikasi berdasarkan model yang telah dipilih.
5. Perbandingan Kinerja

   Sistem menampilkan tabel perbandingan kinerja antara *Naive Bayes* dan SVM, sehingga pengguna dapat melihat mana yang memiliki performa lebih baik.

Untuk memenuhi syarat di atas, komputer harus memenuhi persyaratan fungsional dan non-fungsional. Persyaratan fungsional adalah kebutuhan utama dari perangkat lunak, sementara persyaratan non-fungsional adalah kebutuhan tambahan yang memastikan kinerja yang optimal dari perangkat lunak.

Tabel IV-1. Kebutuhan Fungsional

No	Deskripsi
1	Sistem harus mampu menerima input berupa file audio dalam format umum (.wav) dan mengekstraksi fitur MFCC dari file tersebut.
2	Sistem harus melakukan augmentasi data dengan cara *pitch shifting* dan time stretching untuk meningkatkan variasi dataset.
3	Sistem harus mengimplementasikan dua algoritma klasifikasi, *Naive Bayes* dan SVM, dan memberikan pilihan kepada pengguna untuk menjalankan salah satu atau kedua model sekaligus.
4	Sistem harus mengoptimalkan parameter SVM melalui GridSearchCV untuk menghasilkan model terbaik.
5	Sistem harus menampilkan hasil klasifikasi *genre* dari *music* yang telah diinput.

Tabel IV-2. Kebutuhan Non-Fungsional

No	Deskripsi
1	Sistem harus mampu melakukan pemrosesan file audio dan klasifikasi dalam waktu yang singkat (di bawah 5 detik untuk satu file audio).
2	Sistem harus dapat menangani dataset besar tanpa mengalami penurunan performa yang signifikan, termasuk ketika menggunakan augmentasi data.
3	Antarmuka pengguna harus mudah dipahami oleh pengguna tanpa memerlukan pengetahuan teknis yang mendalam.
4	Sistem harus mampu menangani berbagai jenis file audio dengan format yang berbeda, seperti .wav
5	Perangkat lunak harus dapat dijalankan di berbagai platform sistem operasi seperti Windows, macOS, dan Linux.

4.2.3 Analisis

Pada tahap ini dilakukan analisis terhadap kebutuhan bisnis, kebutuhan fungsional, dan kebutuhan non-fungsional perangkat lunak yang dikembangkan. Tujuan dari proses analisis adalah untuk memastikan bahwa semua persyaratan diidentifikasi dengan jelas sehingga perangkat lunak yang dikembangkan dapat memenuhi harapan pengguna dalam hal kinerja dan kinerja.

4.2.4 Analisis Perangkat Lunak

Dalam analisis perangkat lunak ini, beberapa aspek penting telah diidentifikasi dan dirancang untuk mendukung implementasi kebutuhan fungsional dan non-fungsional yang telah diuraikan sebelumnya. Perangkat lunak akan menggunakan arsitektur berbasis client-server, di mana pengguna (client) dapat mengunggah file audio yang kemudian diproses di server. Proses komputasi berat, seperti ekstraksi fitur *Mel-Frequency Cepstral Coefficients* (MFCC) dan klasifikasi *genre* musik, akan dilakukan di server untuk meningkatkan efisiensi. Model bisnis ini memungkinkan skalabilitas yang lebih baik serta memaksimalkan kemampuan perangkat lunak dalam menangani berbagai ukuran file dan dataset yang besar.

Algoritma yang dipilih untuk klasifikasi *genre* musik adalah *Naive Bayes* dan *Support Vector Machine* (SVM). Kedua algoritma ini terkenal dengan keandalannya dalam tugas-tugas klasifikasi, khususnya yang melibatkan data berbasis pola seperti audio. Selain itu, untuk SVM, akan dilakukan optimisasi parameter menggunakan teknik GridSearchCV. Optimisasi ini bertujuan untuk menemukan parameter terbaik sehingga dapat meningkatkan akurasi model dalam klasifikasi *genre* musik. Dengan adanya dua algoritma ini, perangkat lunak akan memberikan fleksibilitas kepada pengguna untuk memilih model klasifikasi yang sesuai, serta memungkinkan perbandingan kinerja antara keduanya.

Pada tahap ekstraksi fitur audio, sistem akan menggunakan teknologi yang mendukung penggunaan MFCC sebagai representasi numerik dari karakteristik frekuensi audio. Library seperti LibROSA akan digunakan untuk mendukung

proses ini, mengingat LibROSA adalah salah satu library yang andal dan efisien untuk ekstraksi fitur audio. Untuk meningkatkan variasi data dan performa model, teknik augmentasi data seperti *pitch shifting* dan time stretching akan diterapkan. Teknik augmentasi ini membantu sistem dalam menghasilkan lebih banyak variasi data untuk melatih model klasifikasi, sehingga model dapat bekerja lebih baik dalam menangani berbagai variasi *genre* musik.

Dari sisi antarmuka pengguna, fokus utama adalah memberikan kemudahan penggunaan tanpa memerlukan pengetahuan teknis yang mendalam dari pengguna. Antarmuka akan menyediakan fitur-fitur seperti unggahan file audio, pilihan model klasifikasi, serta visualisasi hasil klasifikasi. Visualisasi hasil ini tidak hanya menampilkan *genre* yang terklasifikasi, tetapi juga memberikan perbandingan performa antara model *Naive Bayes* dan SVM. Desain ini diharapkan memberikan pengalaman pengguna yang optimal dan intuitif.

Selain itu, perangkat lunak ini juga dirancang agar cocok dengan beberapa sistem operasi, termasuk Windows, macOS, dan Linux. Fleksibilitas ini penting untuk memastikan perangkat lunak bisa digunakan oleh berbagai jenis pengguna dan sistem operasi yang berbeda. Integrasi teknologi ini mendukung pengembangan perangkat lunak yang cepat, efisien, dan memperhatikan kebutuhan pengguna.

4.3 Fase Design

Pada fase desain, dilakukan perancangan struktur perangkat lunak dan diagram alur data yang bertujuan untuk menggambarkan bagaimana sistem akan bekerja secara keseluruhan. Desain ini mencakup aspek arsitektur perangkat lunak, alur data, interaksi antar komponen, serta perancangan antarmuka pengguna yang diharapkan mampu memenuhi kebutuhan fungsional dan non-fungsional yang telah ditentukan pada fase analisis.

4.3.1 Design Perangkat Lunak

Desain Perangkat lunak digambarkan dalam bentuk *usecase*, flow chart dan data flow diagram.

1. Model *Use Case*

Model UseCase digunakan untuk menggambarkan skenario interaksi pengguna dengan sistem secara detail, mencakup fitur-fitur utama yang disediakan oleh perangkat lunak. Berikut adalah rincian use case untuk sistem klasifikasi *genre* musik:

Diagram use case ini menggambarkan sistem klasifikasi *genre* musik dengan satu aktor utama yaitu User (Pengguna). Sistem ini memiliki tiga fitur utama yang saling terkait:

1. Unggah File Audio: Pengguna dapat mengunggah file audio ke dalam sistem. Proses ini merupakan langkah awal dalam klasifikasi *genre* musik.
2. Ekstraksi Fitur Audio: Fitur ini memiliki relasi <<include>> dengan Unggah File Audio, yang berarti setiap kali file audio diunggah, proses ekstraksi fitur audio akan otomatis dilakukan. Ekstraksi fitur audio adalah langkah penting untuk menganalisis karakteristik suara dari file yang diunggah.
3. Klasifikasi *Genre* Musik: Proses ini juga memiliki relasi <<include>> dengan Ekstraksi Fitur Audio. Ini menunjukkan bahwa klasifikasi *genre* musik bergantung pada hasil ekstraksi fitur audio. Setelah fitur-fitur audio diekstraksi, sistem akan menggunakan informasi tersebut untuk mengklasifikasikan *genre* musik dari file yang diunggah.

Alur kerja sistem:

1. User mengunggah file audio ke sistem.
2. Sistem secara otomatis melakukan ekstraksi fitur audio dari file yang diunggah.
3. Berdasarkan fitur yang diekstraksi, sistem melakukan klasifikasi *genre* musik.

Diagram ini menggambarkan alur kerja dasar di mana pengguna dapat mengunggah file audio untuk diklasifikasikan *genre* musiknya. Ekstraksi fitur audio menjadi langkah kunci yang menghubungkan proses unggah file dengan klasifikasi *genre*, memastikan bahwa sistem memiliki data yang diperlukan untuk melakukan klasifikasi yang akurat.

Tabel IV-3. Skenario *use case* upload file

Identifikasi	
Nomor	1
Tujuan	Mengunggah file audio untuk diklasifikasikan oleh sistem.
Deskripsi	Pengguna mengunggah file audio melalui antarmuka aplikasi untuk memulai proses klasifikasi.
Aktor	User
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pengguna membuka halaman website	
	2. Sistem menampilkan antarmuka untuk unggah file.
3. Pengguna memilih file audio dari perangkat.	
	4. Sistem menerima file audio.
5. Pengguna menekan tombol Upload	
	6. Sistem memverifikasi format file.
7. Jika file tidak valid, pengguna melihat pesan error.	

	8. Sistem menampilkan pesan error.
9. Jika file valid, pengguna menerima konfirmasi.	
	10. Sistem menyimpan file dan memberi tahu bahwa file berhasil diunggah.
Kondisi Akhir	File audio berhasil diunggah dan disimpan oleh sistem.

Tabel IV-4. Skenario *use case* Ekstraksi Fitur Audio

Identifikasi	
Nomor	2
Tujuan	Mengekstraksi fitur audio dari file yang diunggah untuk keperluan klasifikasi.
Deskripsi	Setelah file audio diunggah, sistem secara otomatis mengekstraksi fitur MFCC.
Aktor	Sistem
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pengguna telah mengunggah file audio.	
	2. Sistem menerima file audio yang telah diunggah.
	3. Sistem memulai proses ekstraksi fitur MFCC.
	4. Sistem menyimpan hasil ekstraksi untuk klasifikasi.

Kondisi Akhir	Fitur MFCC berhasil diekstraksi dan siap untuk proses klasifikasi.

Tabel IV-5. Skenario *use case* klasifikasi *genre* musik

Identifikasi	
Nomor	3
Tujuan	Melakukan klasifikasi *genre* musik berdasarkan algoritma yang dipilih.
Deskripsi	Sistem akan memproses file audio dan mengklasifikasikan *genre* musik.
Aktor	User
Skenario Utama	
Aksi Aktor	Reaksi Sistem
1. Pengguna meminta klasifikasi *genre* musik.	
	2. Sistem menerima permintaan klasifikasi.
	3. Sistem menggunakan fitur audio yang telah diekstraksi.
	4. Sistem menjalankan algoritma klasifikasi.
	5. Sistem menentukan *genre* musik berdasarkan hasil klasifikasi.
	6. Sistem menampilkan hasil klasifikasi kepada pengguna.
7. Pengguna melihat hasil klasifikasi *genre* musik. Kondisi Akhir *Genre* musik berhasil	

diklasifikasikan dan hasilnya ditampilkan kepada pengguna.	
Kondisi Akhir	*Genre* musik berhasil diklasifikasikan dan hasilnya disimpan.

4.3.2 Tahapan Proses *Music Genre Classification*

Proses klasifikasi *genre* musik melibatkan beberapa langkah utama, dimulai dari interaksi pengguna hingga hasil klasifikasi ditampilkan. Setiap tahap dalam proses ini dirancang untuk memastikan bahwa file audio yang diunggah dapat diproses secara efektif oleh sistem, menggunakan teknik ekstraksi fitur dan model klasifikasi yang sesuai. Dalam sistem ini, pengguna berperan aktif dalam menentukan input serta memilih model yang akan digunakan dalam proses klasifikasi.

Langkah-langkah dalam proses ini diringkas dalam bentuk *flowchart* yang memberikan gambaran alur kerja sistem secara visual. *Flowchart* ini memudahkan pemahaman mengenai tahapan-tahapan yang dilalui mulai dari input awal hingga output akhir. Setiap langkah penting dijelaskan dengan detail, termasuk keputusan-keputusan yang memengaruhi jalannya proses.

Flowchart ini memberikan gambaran langkah-langkah yang terjadi dari awal pengguna mengunggah file audio hingga hasil klasifikasi *genre* musik ditampilkan. Setiap langkah dalam flowchart ini menunjukkan proses yang dilakukan oleh sistem serta keputusan-keputusan yang dibuat selama proses berlangsung.

- *Start*: Proses dimulai dari titik ini, yang merupakan awal dari keseluruhan alur sistem.

- *Upload File Audio*: Pada langkah ini, pengguna diminta untuk mengunggah file audio yang akan diproses oleh sistem. File ini merupakan input utama yang digunakan dalam proses klasifikasi.
- *Is Format Valid?* : Setelah file audio diunggah, sistem akan memeriksa apakah format file audio yang diunggah valid atau tidak. Jika format file tidak sesuai dengan yang diharapkan, sistem akan kembali ke langkah Upload File Audio, meminta pengguna untuk mengunggah file yang valid. Jika formatnya valid, proses akan dilanjutkan ke langkah berikutnya.
- *Extract Audio Feature*: Setelah file audio valid, sistem akan mengekstraksi fitur audio dari file yang diunggah. Fitur-fitur ini mungkin termasuk Mel-Frequency Cepstral Coefficients (MFCC) atau fitur audio lainnya yang relevan untuk klasifikasi *genre* musik.
- *Classify Music Genre*: Berdasarkan fitur-fitur yang telah diekstraksi, sistem kemudian melakukan proses klasifikasi untuk menentukan *genre* musik dari file audio tersebut. Proses ini menggunakan model klasifikasi yang telah ditentukan sebelumnya.
- *End*: Alur kerja sistem berakhir setelah proses klasifikasi selesai. Meskipun tidak terlihat secara eksplisit dalam flowchart, hasil klasifikasi biasanya akan ditampilkan kepada pengguna sebagai output akhir dari proses ini.

4.3.3 *Data Flow Diagram* (DFD)

Sub bab ini menyajikan *Data Flow Diagram* (DFD) untuk sistem klasifikasi *genre* musik. DFD ini terbagi menjadi dua level: Level 0 dan Level 1. Diagram ini menggambarkan bagaimana data bergerak di dalam sistem, serta interaksi antara pengguna dan berbagai komponen sistem.

1. Level 0 DFD

Pada Level 0, diagram menggambarkan tiga elemen utama:

- *User*: Pengguna yang berinteraksi dengan sistem untuk mengunggah file audio dan melihat hasil klasifikasi.
- *Music Genre Classification System*: Proses inti yang melakukan klasifikasi *genre* musik berdasarkan file audio yang diunggah.
- Output Results: Hasil klasifikasi yang ditampilkan kepada pengguna. Diagram Level 0 menunjukkan alur utama data dari pengguna ke sistem dan dari sistem ke output hasil.

2. Level 1 DFD

Level 1 DFD memberikan rincian lebih lanjut tentang proses yang terjadi di dalam sistem klasifikasi *genre* musik. Terdapat lima proses utama yang dijelaskan:

1. *Upload Audio File* (1.0):
    - Input: User mengunggah audio file ke sistem.
    - Output: Audio file diteruskan ke proses ekstraksi fitur.
2. *Extract Features* (2.0):
    - Input: Menerima audio file dari proses Upload Audio File.
    - Proses: Sistem mengekstrak fitur audio dari file yang diunggah, seperti MFCC (Mel Frequency Cepstral Coefficients).
    - Output: Hasil ekstraksi fitur diteruskan ke proses klasifikasi *genre*.
3. *Classify Genre* (3.0):
    - Input: Menerima hasil ekstraksi fitur dari proses Extract Features.
    - Proses: Sistem mengklasifikasikan *genre* musik berdasarkan fitur yang diekstraksi.
    - Output: Hasil klasifikasi dikirimkan kembali ke User.

Alur Data:

- *User* mengirimkan audio file ke proses Upload Audio File.
- Audio file diteruskan dari Upload Audio File ke Extract Features untuk diekstraksi fiturnya.

- Hasil ekstraksi fitur dikirimkan dari Extract Features ke Classify *Genre* untuk proses klasifikasi.
- Hasil klasifikasi *genre* dikirimkan dari Classify *Genre* kembali ke User.

4.3.4 Perancangan Antarmuka

Perancangan antarmuka pengguna (UI) adalah aspek penting dari pengembangan perangkat lunak karena antarmuka yang baik dapat meningkatkan pengalaman pengguna. Berikut adalah deskripsi dari antarmuka yang akan dirancang untuk aplikasi klasifikasi *genre* musik.

Antarmuka ini menampilkan tombol Upload yang memungkinkan pengguna untuk memilih file audio dari perangkat mereka.Selain itu, terdapat petunjuk mengenai format file yang diterima (.wav) dan batasan ukuran file.

Setelah mengunggah file audio, pengguna akan diarahkan ke halaman pemilihan model klasifikasi. Pengguna dapat memilih antara dua opsi: *Naive Bayes* atau SVM, atau memilih kedua model sekaligus. Terdapat tombol Submit untuk melanjutkan ke proses klasifikasi.

Setelah klasifikasi berhasil, halaman ini akan memperlihatkan hasil dari klasifikasi *genre* musik yang terdeteksi. Jika pengguna memilih untuk membandingkan kedua algoritma, sistem akan menampilkan hasil keduanya.

Antarmuka dirancang agar responsif dan dapat diakses di berbagai perangkat, baik komputer, tablet, maupun smartphone. Tata letak yang intuitif dan navigasi yang mudah digunakan menjadi fokus utama, memastikan bahwa pengguna tidak memerlukan pengetahuan teknis untuk menggunakan aplikasi.

4.4 Fase Coding

Fase coding adalah tahap di mana perangkat lunak yang telah dirancang pada fase sebelumnya akan diimplementasikan dalam bentuk kode sumber. Proses ini melibatkan penulisan, pengujian, dan integrasi komponen sistem untuk memastikan

bahwa semua fungsi berjalan dengan baik sesuai dengan spesifikasi yang telah ditentukan.

4.4.1 Implementasi Antarmuka

Implementasi antarmuka pengguna dilakukan menggunakan Streamlit, yang memungkinkan pengembangan aplikasi web interaktif dengan cepat. Antarmuka dirancang untuk memberikan pengalaman pengguna yang intuitif dan responsif. Berikut adalah elemen utama dalam implementasi antarmuka:

Pada halaman ini, Pengguna dapat mengunggah file audio yang ingin diklasifikasikan melalui antarmuka ini. Sistem akan secara otomatis mengekstrak fitur dari file audio yang diunggah, melakukan preprocessing, dan menampilkan hasil analisis.

Pada halaman ini, Pengguna dapat memilih model klasifikasi yang ingin digunakan, yaitu *Naive Bayes* atau *Support Vector Machine* (SVM). Masing-masing model akan dioptimalkan berdasarkan parameter yang telah ditentukan sebelumnya.

Setelah pengguna memilih model klasifikasi dan memproses file audio, hasil klasifikasi akan ditampilkan. Selain itu, metrik evaluasi seperti akurasi, *precision*, *recall*, dan *F1-score* akan disajikan untuk membantu pengguna memahami performa model yang dipilih.

Antarmuka pengguna ini juga dirancang agar tetap responsif pada berbagai perangkat, termasuk desktop, tablet, dan smartphone. Hal ini dilakukan untuk memastikan bahwa pengguna dapat berinteraksi dengan aplikasi dengan nyaman, terlepas dari jenis perangkat yang digunakan.

4.4.2 Implementasi Ekstraksi Fitur dan Augmentasi

Pada fase ini, fitur *Mel-Frequency Cepstral Coefficients* (MFCC) digunakan sebagai representasi fitur utama dari audio untuk proses klasifikasi *genre* musik. Selain itu, dilakukan augmentasi data menggunakan metode *pitch shifting* dan time stretching untuk meningkatkan variasi dataset yang digunakan oleh model.

1. Ekstraksi Fitur MFCC

   Ekstraksi MFCC dilakukan dengan menghitung rata-rata dari tiga komponen utama:

   1. MFCC: Menangkap representasi frekuensi utama dari sinyal audio.
   2. Delta MFCC: Menggambarkan perubahan temporal dari MFCC (perubahan antar frame).
   3. Delta-Delta MFCC: Menangkap perubahan lebih lanjut dari Delta MFCC.

   Ketiga komponen ini digabungkan menjadi satu array berdimensi 1D dengan panjang 39 fitur untuk setiap file audio. Fitur-fitur ini merepresentasikan karakteristik frekuensi dari sinyal audio yang sangat berguna dalam membedakan *genre* musik. Hasil dari proses ini adalah representasi numerik dari fitur audio dalam bentuk array.

2. Bentuk Data

   Fitur MFCC, Delta, dan Delta-Delta disusun menjadi array 1D. Ukuran array tergantung pada jumlah n_mfcc (biasanya 13) dan penambahan fitur delta

(pertama dan kedua). Dalam hal ini, n_mfcc=13, maka total fitur yang dihasilkan adalah 39 (13 MFCC + 13 Delta + 13 Delta-Delta).

Contoh format data hasil ekstraksi:
[-1.1359882e+02 1.2157067e+02 -1.9162262e+01 4.2363941e+01]

3. Batasan Data

Nilai dari fitur MFCC berada dalam kisaran -100 hingga 100, tergantung pada karakteristik sinyal audio. Nilai ini bergantung pada intensitas frekuensi yang terekstraksi dari audio. Komponen Delta dan Delta-Delta memberikan informasi tambahan mengenai perubahan temporal dari nilai-nilai MFCC, yang penting untuk menangkap dinamika dalam sinyal audio musik.

4. Augmentasi Data

Untuk memperkaya dataset, dua jenis augmentasi diterapkan:

1. Pitch Shifting: Teknik ini mengubah nada dari sinyal audio tanpa mengubah durasinya. Misalnya, meningkatkan nada dua semitone lebih tinggi atau menurunkannya. Hal ini menghasilkan varian audio yang masih relevan secara musik namun dengan pitch yang berbeda.
2. Time Stretching: Teknik ini mengubah kecepatan sinyal audio, baik dengan mempercepat atau memperlambat audio, tanpa mengubah pitch (nada). Dengan cara ini, model akan dilatih pada sinyal dengan durasi yang bervariasi.

Setiap hasil augmentasi juga diekstrak fitur MFCC-nya dengan metode yang sama, sehingga memperluas jumlah data training yang tersedia. Nilai dari fitur MFCC hasil augmentasi tetap berada dalam kisaran yang sama, namun mencerminkan variasi temporal akibat *pitch shifting* atau time stretching.

5. Visualisasi *Waveform* Hasil Augmentasi

Untuk memberikan pemahaman yang lebih mendalam tentang efek augmentasi pada sinyal audio, berikut adalah visualisasi *waveform* dari audio asli dan hasil augmentasinya:

1. Original Audio

   Waveform audio asli menunjukkan:

   - Durasi sekitar 30 detik pada sumbu x.
   - Amplitudo berkisar dari -0,5 hingga 0,5 pada sumbu y.
   - Variasi amplitudo yang mencerminkan perubahan kekerasan dan karakteristik audio.

2. Audio dengan *Pitch Shifting*

   Waveform audio setelah *pitch shifting* (2 semitone) memperlihatkan:

   - Durasi yang sama dengan audio asli (30 detik).
   - Bentuk umum gelombang yang mirip dengan aslinya.
   - Perbedaan halus dalam kepadatan dan frekuensi gelombang, mencerminkan perubahan nada.

3. Audio dengan *Time Stretching*

   Waveform audio setelah *time stretching* (rate 0,8) menunjukkan:

   - Durasi yang lebih panjang, sekitar 37-38 detik.
   - Bentuk gelombang yang mirip dengan aslinya tetapi tersebar dalam periode waktu yang lebih lama.

- Pola gelombang individual yang lebih tersebar, mencerminkan kecepatan pemutaran yang lebih lambat.

6. Visualisasi MFCC

Untuk memberikan gambaran yang lebih jelas mengenai hasil ekstraksi MFCC dan augmentasi, berikut adalah visualisasi dari MFCC asli dan hasil augmentasi:

1. MFCC untuk Audio Asli: Visualisasi ini menunjukkan fitur MFCC yang dihasilkan dari audio asli sebelum augmentasi. Setiap sumbu waktu (x-axis) menampilkan frame audio, dan sumbu frekuensi (y-axis) menampilkan koefisien MFCC.

Gambar ini merupakan representasi fitur Mel-Frequency Cepstral Coefficients (MFCC) dari audio asli, audio yang dipakai ber*genre* blues. MFCC adalah metode yang digunakan untuk menggambarkan karakteristik frekuensi dari sinyal audio. Sumbu **x** menunjukkan waktu dalam detik, dengan setiap kolom mewakili sebuah frame (segmen waktu) dari sinyal audio, dan sumbu **y** menunjukkan koefisien MFCC yang berkaitan dengan frekuensi tertentu. Setiap warna di dalam peta ini mewakili kekuatan atau intensitas frekuensi pada waktu tertentu—semakin merah menunjukkan intensitas positif yang lebih tinggi, sedangkan semakin biru menunjukkan intensitas negatif atau rendah. Daerah di bagian bawah peta MFCC menunjukkan frekuensi rendah, yang biasanya dominan dalam suara bass, sedangkan bagian atas cenderung merepresentasikan frekuensi tinggi yang lebih berhubungan dengan suara instrumen seperti simbal. MFCC asli ini penting sebagai baseline karena akan dibandingkan dengan hasil augmentasi untuk melihat bagaimana fitur audio berubah.

2. MFCC untuk Audio Hasil *Pitch Shifting*: Setelah pitch shifting, hasil MFCC berubah sesuai dengan pergeseran frekuensi dalam sinyal. Visualisasi ini menunjukkan bagaimana fitur MFCC bervariasi dengan adanya perubahan pitch.

Pitch shifting adalah teknik augmentasi yang mengubah nada atau frekuensi sinyal audio tanpa mengubah durasinya. Pada gambar ini, hasil MFCC dari audio yang telah di-*pitch shift* memperlihatkan perubahan yang cukup signifikan pada frekuensi. Misalnya, jika pitch dinaikkan, sebagian besar komponen frekuensi pada gambar ini bergeser ke arah frekuensi yang lebih tinggi, yang ditunjukkan oleh pergeseran pola merah dan biru ke bagian atas. Ini berarti suara terdengar lebih tinggi daripada suara asli. Namun, karena durasi audio tidak berubah, waktu pada sumbu x tetap sama, dengan pola frekuensi yang serupa tetapi dengan perubahan di intensitas dan distribusi frekuensi. Visualisasi ini memberikan wawasan tentang bagaimana frekuensi audio dapat berubah dengan *pitch shifting*, yang berguna untuk memperkaya variasi data saat melatih model.

3. MFCC untuk Audio Hasil Time Stretching: Pada audio yang mengalami time stretching, perubahan durasi mempengaruhi nilai MFCC yang terekstraksi, terutama dalam segi dinamika temporal. Visualisasi ini memperlihatkan bagaimana fitur berubah karena perubahan kecepatan audio.

Time stretching adalah teknik augmentasi lain yang mengubah kecepatan (durasi) sinyal audio tanpa mengubah frekuensi (*pitch*). Gambar ini menunjukkan bagaimana MFCC berubah ketika audio diperlambat (*time-stretching*). Pada sumbu x, pola koefisien MFCC terlihat lebih lebar, yang berarti frame audio diperpanjang. Setiap frame membutuhkan waktu lebih lama untuk selesai dibandingkan audio asli, sehingga sumbu waktu terlihat meluas. Namun, distribusi frekuensi pada sumbu y masih sama seperti audio asli, karena pitch (nada) tidak berubah. Visualisasi ini memperlihatkan efek dari perubahan temporal terhadap sinyal audio. Dalam konteks augmentasi,

time stretching menambah keragaman temporal dalam dataset dengan mempertahankan karakteristik frekuensi sambil memperpanjang atau memperpendek durasi audio.

Detail Tambahan

- Warna pada peta MFCC: Setiap blok warna dalam visualisasi MFCC menggambarkan intensitas energi pada frekuensi tertentu di frame waktu tersebut. Semakin merah atau biru intensitas warna, semakin kuat representasi frekuensi tersebut di titik waktu tersebut. Misalnya, frekuensi rendah yang diwakili oleh blok biru tua atau merah tua menunjukkan suara yang sangat signifikan di frekuensi rendah.

- Perbedaan antara audio asli dan augmentasi: Pada audio asli (Gambar IV 13), frekuensi tetap berada pada posisi aslinya, sementara pada *pitch shifting* (Gambar IV-14), pola koefisien MFCC bergeser ke atas, menandakan pergeseran ke frekuensi yang lebih tinggi. Sedangkan pada *time stretching* (Gambar IV-15), visualisasi terlihat lebih lebar, menandakan bahwa durasi setiap frame diperpanjang tanpa mengubah posisi frekuensi pada sumbu y.

- Pentingnya visualisasi MFCC dalam klasifikasi musik: MFCC berguna karena frekuensi musik cenderung memiliki pola tertentu yang berhubungan dengan *genre* tertentu. Misalnya, musik jazz mungkin memiliki intensitas yang lebih kuat pada frekuensi rendah, sementara musik rock mungkin menunjukkan frekuensi tinggi yang dominan. Dengan augmentasi, model akan belajar mengenali variasi dalam data yang dihasilkan oleh *pitch shifting* dan *time stretching*, membuatnya lebih *robust* dan dapat menangani berbagai variasi dalam dataset.

4.5 Fase Testing

Fase testing adalah tahap penting dalam pengembangan perangkat lunak untuk memastikan bahwa aplikasi berfungsi sesuai dengan spesifikasi yang telah ditetapkan dan bebas dari bug. Pada fase ini, beberapa jenis pengujian dilakukan untuk memvalidasi sistem.

4.5.1 Pemodelan Bisnis

Pemodelan bisnis dalam konteks pengujian bermaksud untuk memastikan agar aplikasi memenuhi kebutuhan pengguna dan tujuan bisnis yang telah ditetapkan. Pada tahap ini, pengujian dijalankan dengan metode *black box*. Metode ini memungkinkan penguji untuk fokus pada input dan output sistem tanpa perlu memperhatikan bagaimana hasil tersebut dicapai.

4.5.2 Kebutuhan Sistem

Kebutuhan sistem dalam konteks pengujian ini mencakup spesifikasi perangkat keras dan perangkat lunak yang digunakan untuk mengembangkan dan menjalankan aplikasi. Sistem pengujian yang ada sebagai berikut:

a. Perangkat Keras:
    a. Laptop: MacBook Air M1
    b. RAM: 8 GB
    c. Penyimpanan: 256 GB
b. Sistem Operasi:
    a. Versi: macOS 15.1 Sequoia
c. Perangkat Lunak:
    a. IDE yang digunakan: Visual Studio Code

4.5.3 Pengujian (*black box*)

Tabel IV-6. Pengujian Dengan *Black Box* Testing

No.	Nama Fitur	Skenario Uji	Input	Output yang Diharapkan	Status
1	Upload File Audio	Pengguna mengunggah	File audio dalam	Sistem menerima dan memverifikasi	Berhasil

		file audio melalui antarmuka.	format .wav	file, serta menampilkan pesan berhasil jika format file valid.	
2	Upload File Audio	Pengguna mengunggah file audio dengan format yang tidak didukung.	File audio dengan format .mp3 atau .aac	Sistem menampilkan pesan error yang menyatakan format file tidak valid.	Berhasil
3	Upload File Audio	Pengguna mengunggah file audio dengan ukuran yang terlalu besar.	File audio melebihi batas ukuran yang ditetapkan	Sistem menampilkan pesan error yang menyatakan file terlalu besar.	Berhasil
4	Ekstraksi Fitur Audio	Sistem mengekstraksi fitur audio dari file yang diunggah menggunakan MFCC.	File audio valid	Fitur audio berhasil diekstraksi dan disimpan untuk proses klasifikasi.	Berhasil
5	Pemilihan Model Klasifikasi	Pengguna memilih algoritma *Naive Bayes* untuk klasifikasi.	Algoritma *Naive Bayes*	Sistem memulai proses klasifikasi menggunakan *Naive Bayes* dan menyimpan hasilnya.	Berhasil
6	Pemilihan Model Klasifikasi	Pengguna memilih algoritma SVM untuk klasifikasi.	Algoritma SVM	Sistem memulai proses klasifikasi menggunakan SVM dan menyimpan hasilnya.	Berhasil
7	Pemilihan Model Klasifikasi	Pengguna memilih kedua algoritma (*Naive Bayes* dan SVM) untuk klasifikasi sekaligus.	*Naive Bayes* dan SVM	Sistem menjalankan kedua algoritma secara bersamaan dan menyimpan hasil keduanya.	Berhasil
8	Klasifikasi *Genre* Musik	Sistem melakukan klasifikasi	File audio valid	Sistem mengklasifikasikan *genre* musik dan	Berhasil

		genre musik berdasarkan fitur yang diekstraksi menggunakan *Naive Bayes* atau SVM.		menampilkan hasil prediksi.	
9	Hasil Klasifikasi	Sistem menampilkan hasil klasifikasi setelah proses selesai.	Hasil prediksi dari *Naive Bayes* atau SVM	Sistem menampilkan *genre* musik yang terprediksi	Berhasil
10	Respons Antarmuka	Pengguna mengakses aplikasi melalui berbagai perangkat (komputer, tablet, smartphone).	Antarmuka di berbagai perangkat	Sistem menampilkan antarmuka yang responsif dan sesuai dengan ukuran layar perangkat.).	Berhasil
11	Time Processing	Sistem memproses file audio dan klasifikasi dalam waktu kurang dari 20 detik.	File audio valid	Sistem menyelesaikan proses klasifikasi dalam waktu maksimal 20 detik per file audio.	Berhasil
12	Kompatibilitas Platform	Sistem diuji di berbagai sistem operasi (Windows, macOS, Linux).	Akses aplikasi dari berbagai sistem operasi	Sistem dapat berjalan lancar di berbagai platform tanpa error.	Berhasil
13	Augmentasi Data	Sistem melakukan augmentasi data (pitch shifting, time stretching) sebelum klasifikasi dilakukan.	File audio valid	Sistem berhasil melakukan augmentasi data dan menggunakan data augmentasi untuk klasifikasi.	Berhasil
14	Error Handling	Pengguna mengunggah	File audio tidak valid	Sistem menampilkan	Berhasil

		file audio dengan format atau ukuran yang tidak valid.		pesan error dengan informasi yang jelas tentang kesalahan format atau ukuran file.	
15	Optimasi Parameter SVM	Sistem melakukan optimasi parameter untuk SVM menggunakan GridSearchCV.	File audio valid, parameter SVM	Sistem berhasil melakukan optimasi parameter dan menghasilkan model terbaik untuk SVM.	Berhasil

4.6 Fase Pemeliharaan

Fase pemeliharaan merupakan tahap krusial dalam siklus hidup perangkat lunak, di mana sistem yang telah di kembangkan dan diuji perlu terus dipantau dan diperbarui agar tetap berkualitas, efisien, dan relevan dengan kebutuhan pengguna. Setelah peluncuran aplikasi klasifikasi *genre* musik, tim pengembang bertanggung jawab untuk melakukan perbaikan bug yang mungkin muncul setelah penggunaan nyata. Masalah yang tidak terdeteksi selama fase pengujian dapat muncul ketika aplikasi dihadapkan pada data atau skenario yang lebih kompleks, sehingga diperlukan pendekatan proaktif untuk mengidentifikasi dan memperbaiki masalah tersebut berdasarkan umpan balik dari pengguna. Selain itu, pembaruan fitur menjadi bagian penting dari fase pemeliharaan ini; dengan terus mendengarkan umpan balik pengguna dan mengikuti perkembangan teknologi, fitur baru dapat ditambahkan atau fitur yang sudah ada dapat ditingkatkan untuk meningkatkan pengalaman pengguna dan efisiensi sistem. Optimasi kinerja juga menjadi perhatian utama, terutama dalam memastikan aplikasi mampu menangani dataset besar dan permintaan pengguna yang tinggi tanpa mengorbankan kecepatan dan responsivitas.

Dukungan pengguna juga menjadi elemen penting dalam fase pemeliharaan, yang mencakup penyediaan dokumentasi yang jelas dan panduan penggunaan serta layanan bantuan untuk menjawab pertanyaan dan mengatasi masalah yang dihadapi

pengguna. Ini tidak hanya meningkatkan pengalaman pengguna tetapi juga membantu dalam membangun hubungan positif antara pengguna dan pengembang. Selain itu, monitoring sistem secara berkala sangat penting untuk memastikan bahwa aplikasi tetap aman dan berfungsi dengan baik. Ini mencakup audit keamanan yang rutin untuk mendeteksi dan mengatasi potensi ancaman, serta analisis kinerja untuk mengidentifikasi area yang perlu dioptimalkan. Dengan semua langkah ini, fase pemeliharaan bertujuan untuk memastikan bahwa aplikasi klasifikasi *genre* musik tidak hanya berfungsi dengan baik pada saat peluncuran, tetapi juga tetap relevan, aman, dan efisien dalam jangka panjang, sehingga dapat memenuhi harapan pengguna yang terus berkembang.

4.7 Kesimpulan

Pengembangan perangkat lunak klasifikasi *genre* musik dilakukan secara bertahap menggunakan Metode Waterfall. Setiap langkah dari analisis kebutuhan, desain sistem, implementasi, pengujian, hingga pemeliharaan dilakukan secara sistematis. Dengan desain modular dan antarmuka pengguna berbasis web, perangkat lunak ini diharapkan mampu memberikan analisis performa yang akurat untuk metode *Naive Bayes* dan SVM dalam klasifikasi *genre* musik.

BAB V
HASIL DAN PEMBAHASAN

5.1 Pendahuluan

Pada bab ini dibahas hasil eksperimen dan evaluasi kinerja dua model klasifikasi yang dipakai dalam penelitian, yaitu *Naive Bayes* dan *Support Vector Machine* (SVM). Kedua model tersebut diuji menggunakan dataset GTZAN yang berisi file audio dari berbagai *genre* musik. Performa model dinilai menggunakan sejumlah metrik, seperti akurasi, presisi, perolehan, dan skor F1. Hasil peramalan dibandingkan dan dianalisis untuk mengetahui kekuatan serta kelemahan masing-masing metode. Selain itu, matriks ketidakpastian disajikan untuk mendapatkan pemahaman lebih dalam tentang sebaran kesalahan prediksi antar spesies.

5.2 Data Hasil Penelitian

Pada bagian ini, hasil evaluasi dari kedua model dipaparkan secara rinci, termasuk perbandingan performa *Naive Bayes* dan SVM berdasarkan metrik yang telah disebutkan sebelumnya.

5.2.1 Konfigurasi Percobaan

Dalam penelitian ini, saya melakukan lima belas konfigurasi percobaan yang berbeda untuk membandingkan performa *Naive Bayes* dan SVM dalam klasifikasi *genre* musik. Setiap konfigurasi memiliki parameter yang berbeda.

Sebelum menjelaskan konfigurasi percobaan, penting untuk memahami komposisi dan pembagian dataset yang digunakan:

- Jumlah total sampel: 1000
- Jumlah *genre* musik (kategori): 10
- Sampel per *genre*: 100

Dataset ini dibagi menjadi set pelatihan (*training set*) dan set pengujian (*test set*) dengan rasio 80:20, menggunakan metode stratified sampling:

- Set pelatihan: 800 sampel (80 sampel per *genre*)
- Set pengujian: 200 sampel (20 sampel per *genre*)

Stratified sampling diterapkan untuk memastikan bahwa proporsi sampel untuk setiap *genre* dipertahankan dalam kedua set. Hal ini penting untuk memastikan evaluasi yang seimbang dan representatif untuk semua *genre*.

Tabel V-1. Konfigurasi Percobaan

Konfigurasi	Model	Parameter
I	*Naive Bayes*	*var_smoothing*:1e-9 (default)
II	*Naive Bayes*	*var_smoothing*:1e-2
III	*Naive Bayes*	*var_smoothing*:1e-5
IV	SVM	*kernel*: rbf , C: 1.0, *gamma*: 'scale' (default)
V	SVM	*kernel*: rbf , C: 10, *gamma*: 0.1
VI	SVM	*kernel*: 'poly', C: 1, degree: 3
VII	SVM	*kernel*: 'sigmoid', C: 1, *gamma*: 'auto'
VIII	SVM	*kernel*: rbf, C: 10, *gamma*: 0.07
IX	SVM	*kernel*: rbf, C: 100, *gamma*: 1

5.2.2 Hasil Konfigurasi I

Naive Bayes dengan parameter default:

Pada konfigurasi pertama ini, model *Naive Bayes* dijalankan menggunakan parameter default, khususnya dengan nilai *var_smoothing* sebesar 1e-9. Parameter *var_smoothing* adalah teknik yang digunakan untuk menangani masalah probabilitas nol dalam model *Naive Bayes*, terutama ketika berurusan dengan fitur yang jarang muncul dalam dataset.

Hasil yang diperoleh dari konfigurasi ini adalah sebagai berikut

Tabel V-2. Hasil konfigurasi I

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*
Naive Bayes	49,25%	0,484821	0,4925	0,482875

Pada Gambar V-1. *Confusion Matrix* Konfigurasi I menunjukkan kinerja model *Naive Bayes* dalam mengklasifikasikan *genre* musik berdasarkan *Confusion Matrix* yang dihasilkan. Pada diagonal utama, terlihat jumlah prediksi yang benar untuk setiap *genre*, seperti 65 instance yang diklasifikasikan dengan benar untuk *genre classical*, 37 untuk *hiphop*, dan 71 untuk *metal*. Nilai yang lebih tinggi pada diagonal ini menunjukkan bahwa model bekerja dengan baik dalam mengklasifikasikan *genre-genre* tersebut. Namun, terdapat juga instance yang salah klasifikasi, yang ditunjukkan oleh nilai-nilai di luar diagonal. Misalnya, 21 instance *blues* salah diklasifikasikan sebagai *disco* dan 12 instance *country* juga salah diklasifikasikan sebagai *disco*.

Dari *Confusion Matrix* ini, terlihat bahwa meskipun model *Naive Bayes* mampu mengenali beberapa *genre* dengan baik, seperti *classical* dan *metal*, model ini masih kesulitan dalam mengklasifikasikan *genre* lain, seperti *blues* dan *reggae*, yang sering kali salah diklasifikasikan sebagai *genre* lain. Misalnya, banyak instance *blues* diklasifikasikan sebagai *disco* dan *reggae* sering salah diklasifikasikan sebagai *rock* atau *disco*.

Hasil performa dari model ini mencakup nilai akurasi sebesar 49,25%, yang berarti sekitar setengah dari seluruh instance berhasil diklasifikasikan dengan benar. *Precision* rata-rata sebesar 0,4848 menunjukkan bahwa 48,48% dari prediksi *genre* yang dihasilkan model sesuai dengan *genre* yang sebenarnya. *Recall* sebesar 0,4925 menunjukkan bahwa model berhasil menangkap 49,25% dari *genre* yang benar, sementara *F1-score* sebesar 0,4829 menunjukkan keseimbangan antara *precision* dan *recall*. Berdasarkan hasil ini, meskipun model *Naive Bayes* memiliki performa yang cukup baik pada beberapa *genre*, masih diperlukan peningkatan, terutama untuk mengatasi kesalahan klasifikasi pada *genre* tertentu.

5.2.3 Hasil Konfigurasi II

Naive Bayes dengan *var_smoothing* yang dioptimalkan:

Pada konfigurasi kedua ini, model *Naive Bayes* dijalankan dengan parameter *var_smoothing* yang telah dioptimalkan. Nilai *var_smoothing* diubah dari default 1e-9 menjadi 1e-2 yang merupakan peningkatan signifikan dalam smoothing yang diterapkan.

Hasil yang diperoleh dari konfigurasi ini adalah sebagai berikut:

Tabel V-3. Hasil Konfigurasi II

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*
Naive Bayes	50,37%	0.499428	0.50375	0.492169

Pada Gambar V-2. *Confusion Matrix* Konfigurasi II, model *Naive Bayes* dioptimalkan dengan mengubah nilai parameter *var_smoothing* dari 1e-9 menjadi 1e-2. Peningkatan nilai *var_smoothing* ini bertujuan untuk meningkatkan stabilitas perhitungan probabilitas, terutama dalam menghadapi fitur-fitur yang jarang muncul dalam dataset. Hasil dari perubahan ini menunjukkan peningkatan kinerja model, meskipun tantangan dalam klasifikasi beberapa *genre* masih terlihat.

Beberapa *genre*, seperti *metal* dan *classical*, masih diklasifikasikan dengan baik, masing-masing dengan 71 dan 65 instance yang diklasifikasikan dengan benar. Namun, beberapa *genre* lain seperti *blues* dan *country* masih sering salah diklasifikasikan. Misalnya, 21 instance *blues* salah diklasifikasikan sebagai *disco*, dan 12 instance *country* juga mengalami kesalahan yang sama. Selain itu, *genre pop* sering diklasifikasikan sebagai *rock*, menunjukkan adanya pola kesalahan yang konsisten dalam klasifikasi beberapa *genre*.

Meskipun terdapat kesalahan klasifikasi, model *Naive Bayes* dengan parameter yang dioptimalkan ini menghasilkan peningkatan akurasi menjadi 50,37%, yang berarti lebih dari separuh instance berhasil diklasifikasikan dengan benar. Selain itu, *precision* model mencapai 0,4994, menunjukkan bahwa hampir 50% dari prediksi yang dihasilkan oleh model sesuai dengan *genre* sebenarnya. *Recall* sebesar 0,5038 mengindikasikan bahwa model mampu mengenali sekitar 50,38% dari *genre* yang benar. Nilai *F1-score* sebesar 0,4922 menunjukkan bahwa keseimbangan antara *precision* dan *recall* berada di sekitar angka 49,22%.

Secara keseluruhan, meskipun peningkatan pada parameter *var_smoothing* memberikan hasil yang lebih baik, terutama dalam hal akurasi, model *Naive Bayes* ini masih menghadapi tantangan dalam menangani beberapa *genre* yang sering mengalami kesalahan klasifikasi, seperti *blues*, *disco*, dan *reggae*. Ini menunjukkan bahwa meskipun ada peningkatan, masih diperlukan penyempurnaan lebih lanjut untuk mencapai hasil klasifikasi yang lebih akurat.

5.2.4 Hasil Konfigurasi III

Naive Bayes dengan *var_smoothing* yang dioptimalkan:

Pada konfigurasi kedua ini, model *Naive Bayes* dijalankan dengan parameter *var_smoothing* yang telah dioptimalkan. Nilai *var_smoothing* diubah dari default 1e-9 menjadi 1e-5, yang merupakan peningkatan signifikan dalam smoothing yang diterapkan.

Hasil yang diperoleh dari konfigurasi ini adalah sebagai berikut:

Tabel V-4. Hasil Konfigurasi III

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*
Naive Bayes	49.625%	0.489267	0.49625	0.486884

Pada Gambar V-3. *Confusion Matrix* Konfigurasi III, model *Naive Bayes* dijalankan dengan nilai *var_smoothing* yang telah dioptimalkan dari 1e-9 menjadi 1e-5. Perubahan ini menghasilkan peningkatan dalam penanganan fitur yang jarang muncul pada dataset, meskipun tantangan dalam klasifikasi *genre* musik masih ada. Pada *Confusion Matrix* ini, terlihat beberapa *genre* yang diklasifikasikan dengan cukup baik, seperti *classical* dengan 64 instance yang diklasifikasikan benar, dan *metal* dengan 69 instance. Namun, beberapa *genre* lain masih menunjukkan kesalahan klasifikasi yang cukup signifikan. Sebagai contoh, *genre blues* memiliki 17 instance yang salah diklasifikasikan sebagai *disco*, sementara *country* dan *reggae* juga mengalami kesalahan klasifikasi yang serupa, masing-masing salah dikenali sebagai *disco* dan *rock*.

Hasil kinerja dari konfigurasi ini mencakup akurasi sebesar 49,625%, yang sedikit lebih baik dari konfigurasi sebelumnya. Selain itu, *precision* model tercatat sebesar 0,4893, yang berarti hampir 48,93% dari prediksi yang dihasilkan oleh model adalah benar. *Recall* sebesar 0,4963 menunjukkan bahwa model berhasil mengenali sekitar 49,63% dari *genre* yang benar. Sementara itu, *F1-score* sebesar 0,4869 menunjukkan keseimbangan antara *precision* dan *recall*.

Secara keseluruhan, meskipun peningkatan parameter *var_smoothing* memberikan sedikit perbaikan dalam akurasi model, beberapa *genre* seperti *blues*, *disco*, dan *reggae* masih menghadapi masalah kesalahan klasifikasi yang cukup tinggi. Hal ini mengindikasikan bahwa meskipun perubahan parameter ini berpengaruh, masih diperlukan peningkatan lebih lanjut pada model atau data preprocessing untuk mencapai hasil klasifikasi yang lebih akurat.

5.2.5 Hasil Konfigurasi IV

SVM dengan parameter default:

Pada konfigurasi ini, model *Support Vector Machine* (SVM) dijalankan menggunakan parameter default, yaitu *kernel* Radial Basis Function (RBF), dengan nilai C sebesar 1.0 dan *gamma* disetel ke 'scale'. *Kernel* RBF digunakan untuk menangkap hubungan non-linear antar fitur, sementara parameter C mengontrol keseimbangan antara memaksimalkan margin dan mengurangi kesalahan klasifikasi. Nilai *gamma* menentukan seberapa jauh pengaruh satu contoh data.

Hasil yang diperoleh dari konfigurasi ini adalah sebagai berikut:

Tabel V-5. Hasil Konfigurasi IV

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*
SVM	82.25%	0.825633	0.82250	0.821801

Pada Gambar V-4. *Confusion Matrix* Konfigurasi IV, model *Support Vector Machine* (SVM) dengan *kernel* Radial Basis Function (RBF) dijalankan menggunakan parameter default, yaitu nilai C sebesar 1.0 dan *gamma* disetel ke 'scale'. Penggunaan *kernel* RBF bertujuan untuk menangkap hubungan non-linear antar fitur, sementara parameter C berfungsi untuk mengatur keseimbangan antara memaksimalkan margin dan mengurangi kesalahan klasifikasi. Nilai *gamma* menentukan seberapa jauh pengaruh satu contoh data terhadap model.

Dari *Confusion Matrix* ini, terlihat bahwa model SVM memiliki performa yang sangat baik dalam mengklasifikasikan berbagai *genre* musik. Misalnya, *genre classical* berhasil diklasifikasikan dengan benar sebanyak 78 instance, *metal*dengan 75 instance, dan *pop* dengan 71 instance. Namun, masih ada beberapa kesalahan klasifikasi, meskipun dalam jumlah yang lebih kecil

dibandingkan dengan model sebelumnya. Sebagai contoh, 10 instance dari *genre blues* salah diklasifikasikan sebagai *reggae*, dan 7 instance dari *country* salah diklasifikasikan sebagai *rock*.

Hasil kinerja model dari konfigurasi ini menunjukkan akurasi sebesar 82.25%, yang jauh lebih baik dibandingkan dengan konfigurasi sebelumnya. Selain itu, *precision* model tercatat sebesar 0,8256, yang menunjukkan bahwa lebih dari 82,56% prediksi yang dihasilkan oleh model adalah benar. *Recall* sebesar 0,8225 mengindikasikan bahwa model berhasil mengenali 82,25% dari *genre* yang benar, dan *F1-score* sebesar 0,8218 menunjukkan keseimbangan yang sangat baik antara *precision* dan *recall*.

Secara keseluruhan, model SVM dengan parameter default ini memberikan hasil yang jauh lebih akurat dalam mengklasifikasikan *genre* musik dibandingkan dengan konfigurasi-konfigurasi sebelumnya. Ini menunjukkan bahwa SVM, dengan *kernel* RBF dan parameter yang diatur secara default, sangat efektif dalam menangani data dengan pola non-linear, sehingga meningkatkan performa klasifikasi secara signifikan.

5.2.6 Hasil Konfigurasi V

SVM dengan parameter yang dioptimalkan:

Pada konfigurasi ini, model *Support Vector Machine* (SVM) dijalankan dengan parameter yang telah dioptimalkan, yaitu *kernel* Radial Basis Function (RBF), dengan nilai C sebesar 10 dan *gamma* sebesar 0.1. Nilai C yang lebih besar memungkinkan model untuk lebih memfokuskan pada pengurangan kesalahan klasifikasi, sementara *gamma* yang lebih tinggi mempersempit pengaruh contoh data, sehingga model lebih responsif terhadap variasi data.

Hasil yang diperoleh dari konfigurasi ini adalah sebagai berikut:

Tabel V-6. Hasil Konfigurasi V

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*
SVM	94.875%	0.950092	0.94875	0.948828

Pada Gambar V-5. *Confusion Matrix* Konfigurasi V, model *Support Vector Machine* (SVM) dijalankan dengan parameter yang telah dioptimalkan, yaitu *kernel* Radial Basis Function (RBF), dengan nilai C sebesar 10 dan *gamma* sebesar 0.1. Parameter C yang lebih tinggi memungkinkan model untuk meminimalkan kesalahan klasifikasi secara lebih agresif, sedangkan *gamma* yang lebih besar mempersempit pengaruh contoh data, sehingga model menjadi lebih sensitif terhadap variasi dalam dataset.

Dari *Confusion Matrix* ini, terlihat bahwa performa model SVM yang dioptimalkan jauh lebih baik dibandingkan dengan konfigurasi sebelumnya. Sebagian besar *genre* musik diklasifikasikan dengan benar, seperti *blues* dengan 75 instance yang diklasifikasikan benar, *classical* dengan 77 instance, dan *pop* dengan 79 instance. Kesalahan klasifikasi masih ada, tetapi sangat minim, seperti 2 instance *blues* yang salah diklasifikasikan sebagai *rock* dan 3 instance *disco* yang salah diklasifikasikan sebagai *reggae*.

Kinerja model ini sangat baik, dengan akurasi mencapai 94.875%, yang menunjukkan bahwa hampir semua instance diklasifikasikan dengan benar. Selain itu, *precision* tercatat sebesar 0,9501, yang berarti bahwa lebih dari 95% dari prediksi yang dibuat oleh model adalah benar. *Recall* sebesar 0,9488 mengindikasikan bahwa model berhasil mengenali sekitar 94,88% dari instance yang benar. Nilai *F1-score* yang sebesar 0,9488 menunjukkan bahwa keseimbangan antara *precision* dan *recall* sangat baik.

Secara keseluruhan, model SVM dengan parameter yang dioptimalkan ini memberikan hasil yang sangat akurat dalam klasifikasi *genre* musik. Peningkatan pada nilai C dan *gamma* terbukti efektif dalam meningkatkan performa model, menjadikan model ini jauh lebih mampu menangani variasi dalam dataset dan menghasilkan klasifikasi yang jauh lebih tepat.

5.2.7 Hasil Konfigurasi VI

SVM dengan *kernel* polynomial:

Pada konfigurasi ini, model *Support Vector Machine* (SVM) dijalankan dengan menggunakan *kernel* polynomial, dengan nilai C sebesar 1 dan degree sebesar 3. *Kernel* polynomial digunakan untuk menangkap pola non linear yang dapat dimodelkan sebagai polinomial, sedangkan parameter degree menentukan derajat polinomial yang digunakan dalam keputusan boundary. Nilai C tetap pada 1, menjaga keseimbangan antara margin yang maksimal dan tingkat kesalahan klasifikasi.

Hasil yang diperoleh dari konfigurasi ini adalah sebagai berikut:

Tabel V-7. Hasil Konfigurasi VI

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*
SVM	78.625%	0.831261	0.78625	0.795615

Pada Gambar V-6. *Confusion Matrix* Konfigurasi VI, model *Support Vector Machine* (SVM) dijalankan menggunakan *kernel* polynomial dengan nilai C sebesar 1 dan degree sebesar 3. *Kernel* polynomial digunakan untuk menangkap pola non-linear yang dapat dimodelkan dengan polinomial, sementara degree menentukan kompleksitas polinomial yang digunakan untuk menentukan decision boundary. Nilai C yang dipertahankan pada 1 menjaga keseimbangan antara margin yang optimal dan tingkat kesalahan klasifikasi.

Dari *Confusion Matrix* ini, terlihat bahwa model SVM dengan *kernel* polynomial menunjukkan performa yang cukup baik dalam mengklasifikasikan berbagai *genre* musik. Beberapa *genre* diklasifikasikan dengan baik, seperti *classical* dengan 69 instance yang diklasifikasikan benar, *pop* dengan 68 instance, dan *jazz* dengan 69 instance yang benar. Namun, terdapat beberapa

kesalahan klasifikasi yang lebih sering terjadi pada *genre* tertentu, seperti *rock* yang mengalami kesalahan klasifikasi dengan 15 instance yang salah dikenali sebagai *country*, dan 10 instance *metal* yang salah dikenali sebagai *reggae*.

Hasil performa model menunjukkan akurasi sebesar 78.625%, yang masih tergolong baik meskipun tidak seakurat model dengan *kernel* RBF yang dioptimalkan. Selain itu, *precision* model tercatat sebesar 0,8313, menunjukkan bahwa sekitar 83,13% prediksi yang dihasilkan oleh model adalah benar. *Recall* sebesar 0,7863 mengindikasikan bahwa model mampu mengenali sekitar 78,63% instance yang benar. Sementara itu, *F1-score* sebesar 0,7956 menunjukkan keseimbangan yang baik antara *precision* dan *recall*.

Secara keseluruhan, penggunaan *kernel* polynomial memberikan hasil yang baik dalam menangani data dengan pola non-linear, meskipun masih terdapat beberapa kesalahan klasifikasi yang perlu diperbaiki, terutama pada *genre* seperti *rock*dan *metal*. Model SVM dengan *kernel* polynomial ini mampu menangkap pola yang lebih kompleks, tetapi mungkin tidak sekuat *kernel* RBF dalam hal akurasi klasifikasi *genre* musik secara keseluruhan.

5.2.8 Hasil Konfigurasi VII

SVM dengan *kernel* sigmoid:

Pada konfigurasi ini, model *Support Vector Machine* (SVM) dijalankan menggunakan *kernel* sigmoid, dengan nilai Csebesar 1 dan *gamma* disetel ke 'auto'. *Kernel* sigmoid digunakan untuk mengaproksimasi fungsi aktivasi seperti dalam jaringan saraf, dengan tujuan menangkap pola non-linear dalam data. Nilai *gamma* 'auto' menentukan bahwa parameter *gamma* dihitung berdasarkan jumlah fitur dalam dataset, dan C sebesar 1 menjaga keseimbangan antara margin dan kesalahan klasifikasi.

Hasil yang diperoleh dari konfigurasi ini adalah sebagai berikut:

Tabel V-8. Hasil Konfigurasi VII

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*

SVM	35.5%	0.339001	0.35500	0.342278

Pada Gambar V-7. *Confusion Matrix* Konfigurasi VII, model *Support Vector Machine* (SVM) dijalankan menggunakan *kernel* sigmoid dengan nilai C sebesar 1 dan *gamma* yang disetel ke 'auto'. *Kernel* sigmoid digunakan untuk mengaproksimasi fungsi aktivasi seperti dalam jaringan saraf, dengan tujuan menangkap pola non-linear dalam data. Nilai *gamma* 'auto' memastikan bahwa parameter *gamma* dihitung berdasarkan jumlah fitur dalam dataset, sementara C sebesar 1 menjaga keseimbangan antara margin maksimal dan tingkat kesalahan klasifikasi.

Dari *Confusion Matrix* ini, terlihat bahwa model SVM dengan *kernel* sigmoid memiliki performa yang jauh lebih rendah dibandingkan dengan konfigurasi-konfigurasi sebelumnya. Beberapa *genre* mengalami kesalahan klasifikasi yang signifikan. Misalnya, *genre rock* dan *blues* sering salah diklasifikasikan sebagai *genre* lain, dengan masing-masing 21 instance *rock* dan 15 instance *blues* salah dikenali. Demikian pula, *genre* seperti *country* dan *reggae* menunjukkan kesalahan klasifikasi yang tinggi, dengan banyak instance dari *genre* lain yang keliru diklasifikasikan sebagai *country* atau *reggae*.

Hasil performa model dari konfigurasi ini menunjukkan akurasi yang rendah, yaitu 35.5%, yang berarti model hanya mampu mengklasifikasikan sekitar 35,5% dari instance dengan benar. Selain itu, *precision* model tercatat sebesar 0,339, yang menunjukkan bahwa hanya sekitar 33,9% prediksi model yang benar. *Recall* sebesar 0,355 menunjukkan bahwa model mampu mengenali sekitar 35,5% dari instance yang benar. Sementara itu, *F1-score* sebesar 0,342 menunjukkan bahwa keseimbangan antara *precision* dan *recall* masih tergolong rendah.

Secara keseluruhan, penggunaan *kernel* sigmoid dalam konfigurasi ini tidak memberikan hasil yang baik untuk klasifikasi *genre* musik. Hal ini kemungkinan disebabkan oleh kemampuan *kernel* sigmoid yang kurang tepat dalam menangkap pola non-linear kompleks dalam dataset ini. *Kernel* sigmoid cenderung digunakan

dalam jaringan saraf dan mungkin tidak optimal untuk model SVM dalam konteks ini, sehingga menyebabkan penurunan performa yang signifikan.

5.2.9 Hasil Konfigurasi VIII

SVM dengan parameter yang dioptimalkan:

Pada konfigurasi ini, model *Support Vector Machine* (SVM) dijalankan dengan parameter yang dioptimalkan, yaitu *kernel* Radial Basis Function (RBF), dengan nilai C sebesar 10 dan *gamma* sebesar 0.07. Parameter C yang tinggi memperkuat fokus model pada pengurangan kesalahan klasifikasi, sementara nilai *gamma* yang lebih kecil dibandingkan konfigurasi sebelumnya memungkinkan model untuk mempertimbangkan pengaruh contoh data dalam radius yang lebih luas.

Hasil yang diperoleh dari konfigurasi ini adalah sebagai berikut:

Tabel V-9. Hasil Konfigurasi VIII

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*
SVM	95.25%	0.953114	0.95250	0.952529

Pada Gambar V-8. *Confusion Matrix* Konfigurasi VIII, model *Support Vector Machine* (SVM) dijalankan dengan parameter yang dioptimalkan, menggunakan *kernel* Radial Basis Function (RBF), dengan nilai C sebesar 10 dan *gamma* sebesar 0.07. Nilai C yang tinggi berfungsi untuk memperkuat fokus model pada pengurangan kesalahan klasifikasi, sementara *gamma* yang lebih kecil dibandingkan dengan konfigurasi sebelumnya memungkinkan model untuk mempertimbangkan pengaruh data dalam radius yang lebih luas, sehingga dapat menangkap pola yang lebih kompleks dalam data.

Dari *Confusion Matrix* ini, terlihat bahwa model SVM yang dioptimalkan memiliki performa yang sangat baik. Sebagian besar *genre* musik diklasifikasikan dengan sangat akurat. Misalnya, *blues* memiliki 75 instance yang diklasifikasikan

benar, *classical* dengan 78 instance, dan *rock* dengan 72 instance yang diklasifikasikan benar. Kesalahan klasifikasi yang terjadi sangat minim, dengan hanya beberapa instance yang salah diklasifikasikan, seperti 1 instance *blues* yang salah dikenali sebagai *reggae* dan 4 instance *rock* yang salah dikenali sebagai *metal*.

Hasil kinerja model dari konfigurasi ini menunjukkan akurasi yang sangat tinggi, yaitu 95.25%, yang menunjukkan bahwa hampir semua instance berhasil diklasifikasikan dengan benar. Selain itu, *precision* model tercatat sebesar 0.9531, yang berarti lebih dari 95,31% prediksi yang dibuat oleh model adalah benar. *Recall* sebesar 0.9525 mengindikasikan bahwa model mampu mengenali sekitar 95,25% dari instance yang benar, dan *F1-score* sebesar 0.9525 menunjukkan keseimbangan yang sangat baik antara *precision* dan *recall*.

Secara keseluruhan, model SVM dengan parameter yang dioptimalkan ini memberikan hasil klasifikasi yang sangat akurat dan konsisten. Peningkatan nilai C dan *gamma* telah terbukti efektif dalam meningkatkan performa model, menjadikannya salah satu konfigurasi terbaik untuk menangani data *genre* musik dengan pola yang lebih kompleks.

5.2.10 Hasil Konfigurasi IX

SVM dengan parameter yang dioptimalkan:

Pada konfigurasi ini, model *Support Vector Machine* (SVM) dijalankan dengan parameter yang dioptimalkan, yaitu *kernel* Radial Basis Function (RBF), dengan nilai C sebesar 100 dan *gamma* sebesar 1. Nilai C yang tinggi mengindikasikan bahwa model sangat berusaha untuk meminimalkan kesalahan klasifikasi, sementara *gamma* yang besar menyebabkan model lebih sensitif terhadap contoh data individual, yang dapat menghasilkan keputusan yang lebih kompleks namun berisiko terhadap *overfitting*.

Hasil yang diperoleh dari konfigurasi ini adalah sebagai berikut:

Tabel V-10. Hasil Konfigurasi IX

Model	*Accuracy*	*Precision*	*Recall*	*F1-score*
SVM	57.375%	0.916775	0.57375	0.646926

Pada Gambar V-9. *Confusion Matrix* Konfigurasi IX, model *Support Vector Machine* (SVM) dijalankan menggunakan *kernel* Radial Basis Function (RBF) dengan nilai C sebesar 100 dan *gamma* sebesar 1. Nilai C yang tinggi menunjukkan bahwa model sangat menekankan pengurangan kesalahan klasifikasi, sedangkan *gamma* yang besar membuat model lebih sensitif terhadap setiap contoh data individual. Hal ini memungkinkan model untuk membuat keputusan yang lebih kompleks, namun dengan risiko *overfitting*.

Dari *Confusion Matrix* ini, terlihat bahwa meskipun model dapat mengklasifikasikan beberapa *genre* dengan benar, seperti *country* dengan 80 instance dan *rock* dengan 52 instance yang benar, ada juga beberapa kesalahan klasifikasi yang signifikan. Misalnya, *genre blues* mengalami kesalahan besar dengan 40 instance yang salah diklasifikasikan sebagai *country*, begitu juga *genre pop* dan *metal* yang menunjukkan kesalahan klasifikasi dengan cukup besar. Hasil performa model menunjukkan akurasi sebesar 57.375%, yang berarti model berhasil mengklasifikasikan lebih dari separuh instance dengan benar, meskipun masih ada ruang yang signifikan untuk perbaikan.

Secara keseluruhan, konfigurasi dengan nilai C yang tinggi dan *gamma* yang besar menghasilkan performa yang bervariasi, dengan kekuatan dalam mengklasifikasikan beberapa *genre* dengan baik, namun menunjukkan kelemahan dalam menangani beberapa kesalahan klasifikasi yang signifikan, terutama dalam *genre* yang lebih sulit dibedakan satu sama lain. Model ini mungkin telah menunjukkan tanda-tanda overfitting, yang menyebabkan akurasi keseluruhan menurun meskipun *precision* cukup tinggi.

5.3 Kumpulan Hasil Penelitian

Dari Hasil uji coba, disajikan hasil akhir dari serangkaian eksperimen yang dilakukan dalam penelitian ini. Setiap konfigurasi model baik untuk *Naive Bayes* maupun *Support Vector Machine* (SVM) dievaluasi berdasarkan empat metrik kinerja utama, yaitu *Accuracy*, *Precision*, *Recall*, dan *F1-score*.

Tabel di bawah ini merangkum hasil dari sembilan konfigurasi yang telah diuji, mencakup variasi parameter untuk kedua model. Pada model *Naive Bayes*, dilakukan pengaturan pada parameter *var_smoothing* untuk melihat pengaruh smoothing terhadap kinerja model. Sementara itu, pada model SVM, berbagai kombinasi *kernel* dan parameter seperti C dan *gamma* dioptimalkan untuk mencapai performa terbaik.

Tabel V-11. Kumpulan Hasil Penelitian

Konfigurasi	Model	Parameter	*Accuracy*	*Precision*	*Recall*	*F1-score*
I	*Naive Bayes*	*var_smoothing*:1e-9 (default)	49,25%	0,484821	0,4925	0,482875
II	*Naive Bayes*	*var_smoothing*:1e-2	50,37%	0.499428	0.50375	0.492169
III	*Naive Bayes*	*var_smoothing*:1e-5	49.625%	0.489267	0.49625	0.486884
Rata Rata Metrik			49,748%	0,491172	0,4975	0,4873
IV	SVM	*kernel*: rbf , C: 1.0, *gamma*: 'scale' (default)	82.25%	0.825633	0.82250	0.821801
V	SVM	*kernel*: rbf , C: 10, *gamma*: 0.1	94.875%	0.950092	0.94875	0.948828
VI	SVM	*kernel*: 'poly', C: 1, degree: 3	78.625%	0.831261	0.78625	0.795615

VII	SVM	*kernel*: 'sigmoid', C: 1, *gamma*: 'auto'	35.5%	0.339001	0.35500	0.342278
VIII	SVM	*kernel*: rbf, C: 10, *gamma*: 0.07	95.25%	0.953114	0.95250	0.952529
IX	SVM	*kernel*: rbf, C: 100, *gamma*: 1	57.375%	0.916775	0.57375	0.646926
Rata Rata Metrik			73,97%	0,491172	0,7397	0,7513

5.4 Analisis Hasil Penelitian

Dari hasil eksperimen yang telah dilakukan, terlihat jelas bahwa performa SVM secara konsisten lebih baik daripada *Naive Bayes* dalam tugas klasifikasi *genre* musik. Analisis per konfigurasi memberikan beberapa temuan penting:

A. Performa *Naive Bayes*:
   a. Konfigurasi I hingga III menunjukkan bahwa *Naive Bayes* dengan berbagai nilai *var_smoothing*menghasilkan akurasi yang relatif rendah, berkisar antara 49,25% hingga 50,37%.
   b. Metrik *precision*, *recall*, dan *F1-score* juga menunjukkan performa yang tidak optimal, dengan nilai *precision* tertinggi hanya 0.50 dan *F1-score* sekitar 0.48.
   c. *Confusion Matrix* untuk *Naive Bayes* mengindikasikan bahwa model ini sering salah mengklasifikasikan *genre-genre* yang memiliki karakteristik akustik serupa. *Naive Bayes* lebih cocok untuk tugas klasifikasi sederhana, tetapi untuk data dengan kompleksitas tinggi seperti *genre* musik, performanya terbatas.
   d. Rata-rata akurasi dari ketiga konfigurasi *Naive Bayes* adalah 49,748%, dengan *precision* sebesar 0,491172, *recall* 0,4975, dan *F1-score* 0,4873.

B. Performa SVM:

a. SVM secara signifikan mengungguli *Naive Bayes*, khususnya pada konfigurasi IV hingga IX. Konfigurasi default SVM (*kernel* RBF, C=1, *gamma*='scale') menghasilkan akurasi sebesar 82,25%, yang jauh lebih baik dibandingkan *Naive Bayes*.

b. Konfigurasi terbaik ditemukan pada konfigurasi VIII dengan *kernel* RBF, C=10, dan *gamma*=0.07, yang menghasilkan akurasi 95,25%. Selain akurasi, *precision*, *recall*, dan *F1-score* pada konfigurasi ini juga sangat tinggi (di atas 0.95), menunjukkan kemampuan SVM untuk mengklasifikasikan *genre* dengan sangat baik.

c. Pada konfigurasi lainnya, seperti konfigurasi VII dengan *kernel* sigmoid, performa SVM turun drastis dengan akurasi 35,5%. Ini menunjukkan bahwa pilihan *kernel* sangat mempengaruhi performa SVM, dan *kernel* sigmoid mungkin kurang cocok untuk tugas klasifikasi *genre* musik.

d. *Confusion Matrix* dari konfigurasi terbaik (V dan VIII) menunjukkan distribusi kesalahan yang minim, menunjukkan kemampuan SVM untuk membedakan *genre-genre* yang memiliki karakteristik serupa secara efektif.

e. Rata-rata akurasi untuk semua konfigurasi SVM adalah 73,97%, dengan *precision* 0,491172, *recall* 0,7397, dan *F1-score* 0,7513.

C. Pengaruh Parameter:

a. Perubahan parameter pada SVM, seperti nilai C dan *gamma*, memiliki dampak signifikan terhadap performa. Nilai C yang lebih tinggi dan *gamma* yang tepat pada *kernel* RBF memberikan hasil yang optimal. Sebaliknya, *kernel* sigmoid (konfigurasi VII) dan *kernel* polinomial (konfigurasi VI) tidak mampu menghasilkan hasil yang optimal, dengan akurasi yang lebih rendah.

b. Pemilihan *kernel* RBF dengan parameter yang dioptimalkan (C=10 dan *gamma*=0.07) terbukti menjadi kombinasi yang paling efektif untuk tugas klasifikasi *genre* musik.

5.5 Kesimpulan

Bab ini berisi hasil dari evaluasi dan analisis performa metode *Naive Bayes* dan SVM dalam klasifikasi *genre* musik. Kesimpulan utama yang diperoleh mencakup perbandingan akurasi kedua metode, pemilihan konfigurasi terbaik, serta interpretasi hasil yang diperoleh dari analisis *Confusion Matrix*. Selain itu, bab ini juga membahas implikasi dari temuan penelitian terkait penggunaan SVM sebagai model yang lebih unggul dalam klasifikasi *genre* musik berbasis fitur audio.

BAB VI
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan hasil eksperimen dan analisis yang dilakukan dalam penelitian ini, beberapa kesimpulan utama dapat ditarik mengenai klasifikasi *genre* musik menggunakan fitur *Mel-Frequency Cepstral Coefficients* (MFCC) dengan dua model klasifikasi: *Naive Bayes* dan *Support Vector Machine* (SVM). Berikut poin-poin kesimpulannya:

1. SVM mengungguli *Naive Bayes* dalam semua aspek performansi Berdasarkan hasil eksperimen, SVM dengan *kernel* RBF (C=10, *gamma*=0.07) memberikan kinerja terbaik dengan akurasi 95,25%. SVM secara konsisten mengungguli *Naive Bayes*, yang hanya mampu mencapai akurasi maksimal 50,37%. Ini menunjukkan kemampuan SVM yang lebih baik dalam menangani data non-linear dan fitur audio yang kompleks.
2. Perbedaan rata-rata performa antara *Naive Bayes* dan SVM Rata-rata akurasi yang dicapai *Naive Bayes* pada semua konfigurasi hanya sebesar 49,748%, sementara rata-rata akurasi SVM pada seluruh konfigurasi mencapai 73,97%. Hal ini menunjukkan bahwa secara keseluruhan, SVM memberikan performa yang lebih stabil dan unggul dalam klasifikasi *genre* musik.
3. SVM lebih konsisten pada metrik evaluasi lain selain akurasi, SVM menunjukkan performa yang konsisten dalam hal *precision*, *recall*, dan *F1-score* di semua konfigurasi terbaiknya, dengan nilai *precision* mencapai 0,95. SVM dapat menghindari kesalahan dalam prediksi lebih baik daripada *Naive Bayes*, memberikan hasil yang lebih seimbang dalam mengklasifikasikan berbagai *genre* musik.

4. Distribusi Kesalahan pada SVM Lebih Terkonsentrasi Berdasarkan analisis *Confusion Matrix*, SVM menunjukkan distribusi kesalahan yang lebih sedikit dan lebih terfokus pada *genre* yang sulit dibedakan, sedangkan *Naive Bayes* menunjukkan distribusi kesalahan yang lebih luas, terutama pada *genre* yang memiliki fitur mirip.
5. Pengaruh Parameter SVM Terhadap Performa Pemilihan parameter yang tepat, seperti nilai C dan *gamma* pada *kernel* RBF, sangat memengaruhi performa SVM. Kombinasi parameter optimal (C=10, *gamma*=0.07) memberikan hasil yang terbaik dalam klasifikasi *genre music*.

6.2 Saran

Berdasarkan kesimpulan yang diambil, terdapat beberapa saran untuk penelitian dan peningkatan lebih lanjut dalam klasifikasi *genre* musik maupun klasifikasi audio secara umum:

1. Penggunaan fitur tambahan fitur MFCC telah terbukti efektif, tetapi penambahan fitur lain seperti chroma, spectral contrast, dan tonnetzbisa membantu meningkatkan akurasi model. Dengan memperkaya representasi data audio, model akan lebih baik dalam membedakan *genre* yang mirip.
2. Eksplorasi model Deep Learning penggunaan *Convolutional Neural Networks* (CNNs) atau *Recurrent Neural Networks* (RNNs) untuk klasifikasi audio bisa menjadi langkah selanjutnya. Model-model ini memiliki kemampuan lebih baik dalam mempelajari pola data yang kompleks dan dapat menangkap hubungan temporal antar fitur audio dengan lebih baik.
3. Teknik augmentasi data yang lebih variatif, seperti *pitch shifting* dan time stretching sudah diterapkan, namun dapat diperluas dengan teknik lain seperti *noise injection* dan *random equalization*. Teknik augmentasi yang lebih variatif akan memperkaya data dan membantu model lebih *robust* terhadap variasi audio yang berbeda.

4. Pengoptimalan *hyperparameter* yang lebih efisien meskipun *Grid Search* telah digunakan, metode optimasi yang lebih canggih seperti *Bayesian Optimization* atau *Random Search* bisa digunakan untuk menemukan kombinasi *hyperparameter* terbaik secara lebih efisien. Teknik ini dapat mempercepat proses *tuning* tanpa mengorbankan akurasi.

Dengan mengikuti saran-saran di atas, diharapkan penelitian lebih lanjut dapat menghasilkan model klasifikasi *genre* musik yang lebih akurat dan *robust*, serta siap untuk diimplementasikan pada aplikasi nyata. Penelitian ini memberikan dasar yang kuat untuk eksplorasi lebih jauh dalam bidang klasifikasi audio menggunakan pembelajaran mesin.

Perbandingan Performa Metode Naive Bayes dan Support Vector Machine dalam Klasifikasi Genre Musik Berdasarkan Ekstraksi Fitur Sinyal Audio Menggunakan Mel-Frequency Cepstral Coefficients (MFCC)

ORIGINALITY REPORT

2%	2%	1%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Sriwijaya University Student Paper	1%
2	assets.researchsquare.com Internet Source	1%
3	ojs.serambimekkah.ac.id Internet Source	1%
4	etd.repository.ugm.ac.id Internet Source	1%

Exclude quotes	On	Exclude matches	< 1%
Exclude bibliography	On		

SURAT KETERANGAN PENGECEKAN SIMILARITY

Saya yang bertanda tangan di bawah ini

Nama	: Kevin Putrayudha Naserwan
Nim	: 09021282126083
Prodi	: Teknik Informatika
Fakultas	: Fakultas Ilmu Komputer

Menyatakan bahwa benar hasil pengecekan similarity Skripsi/~~Tesis/Disertasi/Lap.~~ Penelitian yang berjudul Perbandingan Performa Metode Naive Bayes dan Support Vector Machine dalam Klasifikasi Genre Musik Berdasarkan Ekstraksi Fitur Sinyal Audio Menggunakan Mel-Frequency Cepstral Coefficients (MFCC) adalah 2 %.

Dicek oleh operator *:

1. Dosen Pembimbing
2. UPT Perpustakaan
3. Operatur Fakultas.....

Demikianlah surat keterangan ini saya buat dengan sebenarnya dan dapat saya pertanggung jawabkan.

Indralaya, 21 Oktober 2024

Menyetujui
Dosen pembimbing,

Nama: Kanda Januar Mirasawan, S.Kom, M.T.
NIP: 199001092019031012

Yang menyatakan,

Nama: Kevin Putrayudha Naserwan
NIM: 09021282126083

*Lingkari salah satu jawaban tempat anda melakukan pengecekan Similarity