BAB 2
TINJAUAN PUSTAKA

2.1 Musik

Musik merupakan rangkaian bunyi atau suara yang diatur menjadi suatu paduan harmonis yang dihasilkan dari alat-alat yang dapat menghasilkan irama. Musik mempunyai ritme, melodi, dan harmoni yang memberikan kedalaman dan memungkinkan penggunaan beberapa instrumen atau bunyi-bunyian. Musik adalah seni yang dipadukan sebaik mungkin dengan mengguankan suara ataupun alat musik yang memiliki unsur ritme, melodi, harmoni, dan warna nada [5]. Dari pengertian diatas dapat disimpulkan bahwa musik adalah sebuah pola yang tersusun dari bunyi atau suara, keadaan diam (*sounds and silences*) dalam alur waktu dan ruang tertentu dalam urutan, kombinasi, dan hubungan temporal yang berkesinambungan sehingga mengandung ritme, melodi, warna bunyi, dan keharmonisan yang biasanya dihasilkan oleh alat musik atau suara manusia.

2.1.1 Elemen Musik

Menurut [5] musik disusun oleh elemen-elemen sebagai berikut:

a) *Sound: Pitch, Dynamics*, dan *Tone Color*

Pitch merupakan tinggi rendahnya suara yang kita dengar. *Dynamics* merupakan tingkat keras lembutnya suatu musik. *Tone Color* merupakan kualitas yang membedakan warna musik

b) *Musical instruments*

Merupakan setiap alat selain suara yang menghasilkan bunyi musik.

c) *Rhythm*

Merupakan bagaimana sebuah musik mengalir saat dimainkan. Terdiri dari beberapa aspek yang saling berkaitan:

1) *Beat*, ketukan yang menentukan musik tetap seimbang di dalam satu kesatuan waktu.

2) *Meter*, merupakan pengorganisasian beat yang dibuat dalam bentuk pengelompokan,

3) Aksen dan Sinkopasi, sebuah not yang paling jelas ditekankan dengan memainkannya lebih keras dibandingkan not lainnya disebut juga dengan aksen, sedangkan sinkopasi adalah sebuah efek dimana sebuah not aksen muncul kita tidak hanya mengharapkan satu not saja, dan

4) Tempo, cepat lambatnya beat sebuah lagu.

d) *Music notation*

Merupakan sebuah sistem penulisan dalam musik dimana menunjukkan nada dan irama spesifik yang dapat dikomunikasikan.

e) *Melody*

Merupakan nada tunggal yang disatukan sehingga dapat dikenali keseluruhan.

f) *Harmony*

Merupakan sebuah *chord* dibentuk dan bagaimana mereka mengikuti satu sama lain.

g) *Key*

Merupakan pusat melodi, tidak hanya pusat melodi namun pusat skala dan chord.

h) *Musical texture*

Berhubungan dengan bentuk, struktur, pengaturan, dan koherensi.

i) *Musical form*

Bentuk musik berhubungan dengan bentuk , struktur, pengaturan, dan koherensi.

j) *Performance*

Tugas dari *performer* untuk membawakan simbol yang disampaikan oleh komposer

k) *Musical style*

Merupakan karakteristik dalam memainkan melodi, irama, warna nada, dinamika, harmoni, tekstur, dan bentuk

2.1.2 Genre Musik

Genre musik merupakan mengkarakteristikan sebuah musik dengan

menggunakan melodi, ritim, warna nada, dinamik, harmoni, susunan, dan bentuk [5]. Genre musik yang digunakan pada penelitian kali ini adalah:

a) Musik *Rhythm and Blues* (R&B)

Musik R&B terdiri atas berbagai jenis musik populer yang saling terkait. Musik rhythm and blues yang lebih dikenal dengan musik R&B memiliki beberapa genre-genre, seperti, jump blues, club blues, black rock n' roll, soul, funk, disco dan rap.

b) Musik Pop

Musik ini berkembang di Indonesia sekitar tahun 1960-an dan banyak digemari masyarakat khususnya kaum muda atau remaja. Grup musik pop sering disebut dengan sebutan band yang menggunakan peralatan elektronik atau modern. Instrumen yang wajib ada dalam bentuk grup sederhanannya antara lain, drum, gitar melodi dan *rhythm*, piano, dan bass gitar.

c) Musik *Country*

Musik ini sering disebut juga *Country and Western*, yang merupakan salah satu genre besar pada musik populer terutama di negeri Amerika serikat. Jenis musik modern ini bersumber dari musik rakyat (*folk song*) atau musik tradisional yang berasala dari Appalachia di kawasan pegunungan selatan Amerika Serikat.

d) Musik Jazz

Jazz dapat digambarkan secara umum sebagai musik yang berakar pada improvisasi dan ditandai dengan irama yang lemah, beat yang menenangkan, dan warna nada yang khas dan teknik khusus dalam memainkan.

e) Musik Rock

Musik rock merupakan singkatan dari nama genre musik *rock and roll* yang pertama kali dilontarkan pada tahun 1950-an pada publik Amerika Serikat oleh Alan Freed dalam sebuah siaran radio yang menyiarkan acara musik *rhythm and blues* (R&B) secara rutin. Rock merupakan bentuk musik populer yang biasanya diiringi oleh gitar dan drum. Namun banyak juga gaya musik rock yang menggunakan alat musik seperti organ, piano, atau synthetisizers.

f) Musik Klasik

Musik klasik merupakan istilah luas yang biasanya mengarah pada musik yang dibuat di atau berakar dari tradisi kesenian Barat, musik kristiani, dan musik orkestra, mencakup periode dari sekitar abad ke-9 hingga abad ke-21.

g) Musik Disko

Disko adalah salah satu aliran musik untuk dansa yang berkembang pada tahun 1970-an di klub-klub dansa Amerika Serikat. Disk Jockey di klub dansa memutar serangkaian lagu dari piringan hitam tanpa terputus agar orang dapat terus menari tanpa berhenti. Istilah disko juga dipakai untuk acara, ruang atau gedung tempat orang berdansa/menari diiringi rekaman musik sebagai bentuk hiburan.

h) Musik HipHop

Musik Hip hop atau hip-hop atau musik rap adalah genre musik yang tercipta di Amerika Serikat pada tahun 1970an yang melibatkan teknik rap, yaitu cara berbicara yang disuarakan melalui rima yang ritmis, yang disertai oleh musik latar yang ritmis.

i) Musik Metal

Musik metal adalah sebuah aliran musik rock yang berkembang pada 1970-an dengan akar dari blues rock dan psychedelic rock. Aliran musik ini ditandai dengan distorsi Gitar yang sangat kuat, solo gitar panjang, ketukan cepat, baik disemua instrumentasi alat musiknya. Lirik metal pada umumnya berkaitan dengan maskulinitas dan kejantanan.

j) Musik Reggae

Reggae adalah suatu aliran musik yang awalnya dikembangkan di Jamaika pada akhir era 60-an. Sekalipun kerap digunakan secara luas untuk menyebut hampir segala jenis musik Jamaika, istilah reggae lebih tepatnya merujuk pada gaya musik khusus yang muncul mengikuti perkembangan ska dan o ou já tarocksteady. Reggae berbasis pada gaya ritmis yang bercirikan aksen pada off-beat atau sinkopasi, yang disebut sebagai skank. Pada umumnya reggae memiliki tempo lebih lambat daripada ska maupun rocksteady.

2.2 Klasifikasi Genre Musik

Klasifikasi genre musik merupakan cara pengelompokan suatu musik kedalam genrenya atau jenisnya yang memiliki kemiripan satu sama lain. Pada penelitian kali ini akan digunakan 10 genre musik yaitu blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, rock. Data musik nantinya akan dilatih terlebih dahulu untuk mendapatkan cirinya melalui fitur ekstraksi. Kemudian data latih yang sudah dipersiapkan akan diklasifikasikan oleh sistem kedalam genrenya masing-masing.

2.3 *Extraction Feature*

Extraction feature merupakan proses pengolahan data musik menggunakan beberapa fitur sehingga menghasilkan nilai-nilai yang nantinya akan digunakan untuk proses klasifikasi genre musik. *Extraction feature* pada penelitian ini akan dibagi menjadi 2 bagian. Bagian pertama akan menjelaskan proses ekstraksi fitur permukaan musik dan bagian kedua akan menjelaskan proses ekstraksi fitur ritme.

2.3.1 Fitur Permukaan Musik

Pada penelitian ini, kata permukaan musik digunakan untuk menunjukkan karakteristik dari musik yang berhubungan dengan tekstur, timbre, dan instrumentasi. Statistik dari distribusi *spectral* terhadap waktu dapat digunakan untuk merepresentasikan fitur permukaan musik untuk tujuan *pattern recognition.* Fitur vektor yang digunakan pada penelitian ini untuk tujuan tersebut adalah: (*mean-centroid, mean-rolloff, mean-flux, mean-zerocrossings,std-centroid, std-rollof, std-flux, std-zerocrossings,* dan *lowenergy*) [1]. Tahapan dari fitur permukaan musik adalah:

1. Tiap data yang akan digunakan adalah data musik berformat (.wav) dan berdurasi 5 detik dengan sample rate sebesar 22.050 Hz.
2. Pada tahap *analysis windowing*, data 5 detik akan dibagi menjadi 250 *analysis window* yang berisikan 20 milidetik.
3. Hasil dari proses *analysis window* akan digunakan untuk perhitungan *Fast Fourier Transform* (FFT)

4. Pada proses ini diperlukan nilai *magnitude* dari data audio digital. Nilai tersebut dapat didapat dengan terlebih dahulu menghitung nilai riil dan nilai imajiner dari data audio digital menggunakan FFT.
5. Setelah didapatkan nilai *magnitude* maka dapat dihitung nilai dari fitur permukaan musiknya

2.3.1.1 *Fast Fourier Transform* (FFT)

Fast Fourier Transform adalah suatu algoritma yang digunakan untuk merepresentasikan sinyal dalam domain waktu diskrit dan domain frekuensi. Perhatikan definisi dari FFT:

$$F_{[k]} = \sum_{n=0}^{N-1} f(n)\, e^{-j2\pi kn/N} \quad (2.1)$$

Atau dapat dituliskan dengan:

$$F_{[k]} = \sum_{n=0}^{N-1} f(n)\cos(2\pi \mathrm{kn}/N\ ) - j\sum_{n=0}^{N-1} f(n)\sin\ (2\pi \mathrm{kn}/N) \quad (2.2)$$

Untuk melihat nilai hasil FFT digunakan rumus:

$$M_{[k]} = [R^2 + I^2]^{1/2} \quad (2.3)$$

2.3.1.2 *Centroid*

Spectral Centroid merupakan indikator kecerahan spektrum yang ada dan menunjukkan spektral pusat gravitasi. *Spectral centroid* menunjukkan tingkat kejernihan suara [6]. *Centroid* adalah ukuran dari *spectral brightness. Centroid* memiliki persamaan sebagai berikut:

$$C = \sum_{k=0}^{N-1} kM_{[k]} / \sum_{0}^{N-1} M_{[k]} \quad (2.4)$$

2.3.1.3 *Rolloff*

Spectral Rollof didefinisikan sebagai frekuensi dibawah akumulasi kekuatan spectral yang mencapai persentase nilai tertentu antara 0,85 (85%) atau 0,95 (95%) [6]. *Rolloff* adalah ukuran dari *spectral shape.* Rolloff memiliki persamaan sebagai berikut:

$$\sum_{k=0}^{R} M_{[k]} = 0.85 \sum_{k=0}^{N-1} M_{[k]} \quad (2.5)$$

2.3.1.4 *Flux*

Flux adalah ukuran dari *spectral change. Flux* menunjukkan berapa banyak perubahan spektral dari waktu ke waktu.

$$F= \| M_{[k]} - Mp_{[k]} \| \quad (2.6)$$

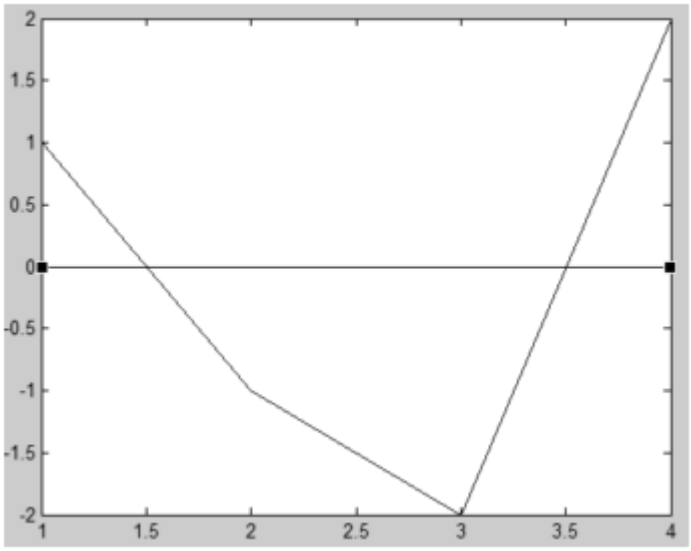
Dengan k= indeks dari *analysis window*. $M_{[k]}$ merupakan nilai magnitude pada $M_{[k]}$ dan $Mp_{[k]}$ nilai magnitude saat k-1. Dalam *flux*, dicari perubahan energy dari satu *window* dengan *window* sebelumnya. Namun pada kasus k=0, energy pada *window* tersebut diurangi dengan nilai 0. Sehingga hasilnya adalah *window* itu sendiri. Selanjutnya pada k=1,…,N-1, *energy window* ke- k akan dikurangi *energy* pada *window* sebelumnya. Kemudian tiap hasil pengurangan akan dikuadradkan. Lalu hasil kuadrat dari tiap titik tersebut akan dijumlahkan. [7].

2.3.1.5 *LowEnergy*

LowEnergy adalah persentase dari jumlah *analysis windows* yang memiliki energi kurang dari rataan seluruh *analysis windows* dalam satu data musik. Misalkan terdapat 250 *analysis windows* dalam satu data musik, rataan seluruh *analysis windows* adalah 500, dan terdapat 80 *analysis windows* yang jumlah energinya kurang dari 500, maka *LowEnergy* adalah persentase dari 80/250.[7]

2.3.1.6 *Zerocrossings*

Zerocrossings adalah jumlah dari penyebrangan sinyal dalam 1 *window* yang melalui nilai 0. Misalnya terdapat sinyal bernilai [1,-1,-2,2] yang dapat dilihat pada gambar 3.6. Dari sinyal tersebut terdapat 2 *Zerocrossings,* yaitu dari nilai 1 ke -1 dan dari nilai -2 ke 2.



Gambar 2. 1 Sinyal Contoh Perhitungan Zerocrossings

2.3.2 Fitur Ritme

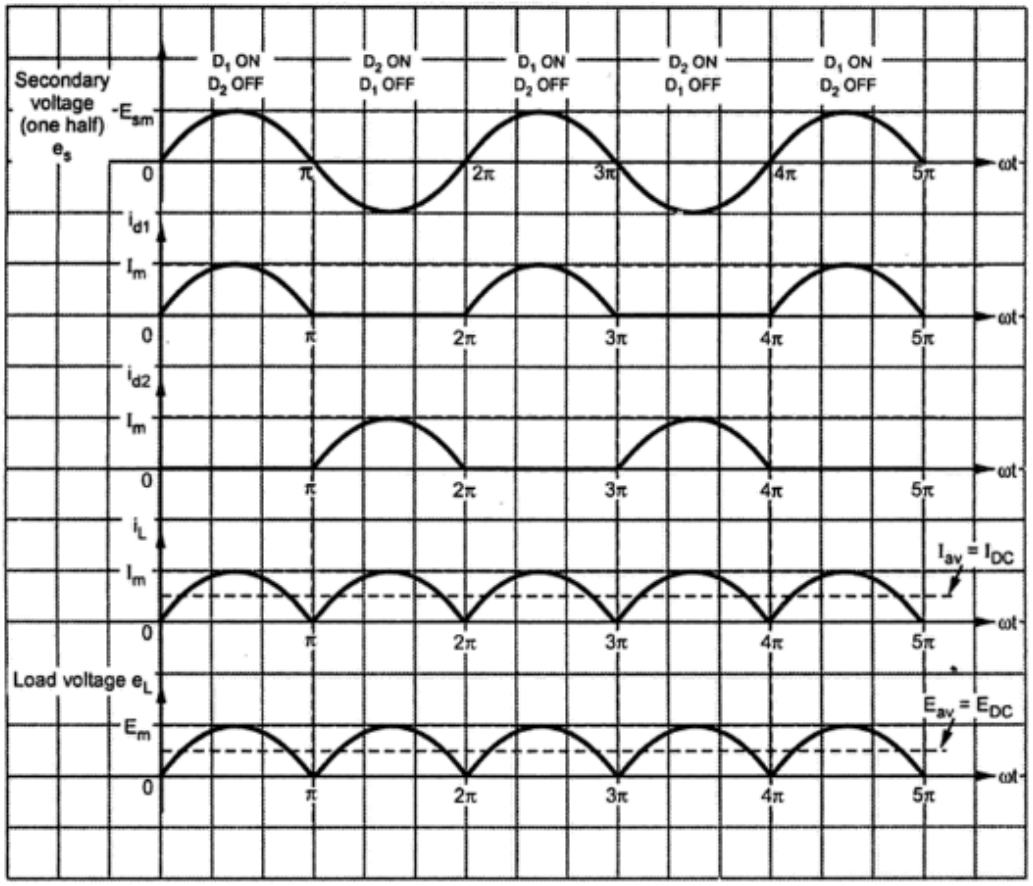
Fitur ritme dikarakterisasikan sebagai pergerakan sinyal musik terhadap waktu dan mengandung informasi seperti yang biasanya terdapat pada ritme, *beat, tempo* dan *time signature*. Fitur yang diset untuk merepresentasikan struktur ritme berasal dari mendeteksi periode yang paling penting pada sebuah sinyal. Fitur ritme dihitung dengan perhitungan *beat histogram* [8].

2.3.2.1 DWT

Wavelet merupakan bentuk pengklasifikasian sinyal dalam versi penskalaan dan pergeseran (*scalling* dan *shifting*) masing-masing dari sinyal sumber atau dikenal dengan *mother wavelet*. Transformasi *wavelet* hampir menyerupai transformasi *fourier* yang merupakan bentuk pengklasifikasian sinyal menjadi bentuk sinusoidal dengan berbagai macam frekuensi[9]. *Discrete Wavelet Transform* (DWT) merupakan salah bentuk diskrit dari *wavelet* dan seringkali digunakan. DWT akan mengubah sinyal sumber menjadi dua klasifikasi sinyal yaitu frekuensi tinggi serta frekuensi rendah. Proses DWT diawali dengan proses *filtering* dengan melewatkan sinyal pada seperangkat *highpass filter* dan *lowpass filter*.Pada penelitian kali ini akan menggunakan DAUB4 sebagai proses filternya.

2.3.2.2 *Full Wave Rectification* (FWR)

Rectifier merupakan sebuah *device* yang merubah tegangan AC menjadi tegangan DC. *Rectifier* terbagi menjadi 2 yaitu *Half Wave Rectifier* (HWR) dan *Full Wave Rectifier* (FWR). Pada HWR nilai yang diambil hanya nilai positifnya saja sedangkan nilai negatifnya akan dihilangkan sedangkan pada FWR nilai negatifnya akan dirubah menjadi positif [10]. Ilustrasi FWR ditunjukan pada gambar 2.2.



Gambar 2. 2 Ilustrasi *Full Wave Rectification*

Full wave rectification (FWR) dihitung dengan persamaan:

$$FWR[n] = abs\ (y[n]) \tag{2.7}$$

2.3.2.3 *Low Pass Filtering* (LPF)

LPF pada penelitian kali ini menggunakan *One Pole Low Pass Filter* dengan α=0.99. Nilai α=0.99 diadopsi dari penelitian tzanetakis[1]. *Low pass filtering* (LPF) dihitung menggunakan persamaan:

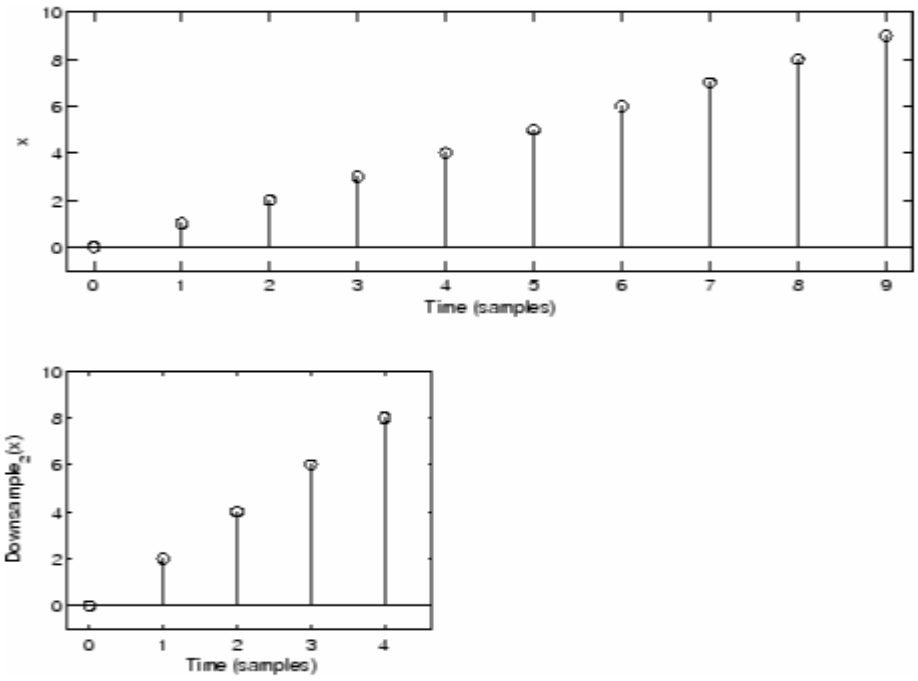
$$LPF[n] = (1-\alpha)\, FWR[n] - \alpha FWR[n] \quad (2.8)$$

2.3.2.4 *Downsampling*

Downsampling (↓) merupakan proses menurunkan banyaknya sampel dari sinyal dengan cara hanya mengambil sampel sinyal pada selang waktu tertentu k. *Downsampling* dihitung dengan persamaan :

$$D[n] = LPF[kn] \quad (2.9)$$

dengan D[n] adalah sinyal keluaran hasil *downsampling* dan LPF[n] adalah nilai masukan yang didapatkan dari proses *Low Pass Filtering.*Berikut adalah ilustrasi *downsampling* dengan k=2 seperti yang ditunjukkan pada gambar 2.3.



Gambar 2. 3 Ilustrasi Downsampling

2.3.2.5 *Normalization*

Normalization (NR) / *mean removal* merupakan proses untuk mengurangi nilai dengan nilai rata-ratanya. NR dihitung menggunakan persamaan:

$$NR[n] = D[n] - E\,D[n] \tag{2.10}$$

2.3.2.6 *Autocorrelation*

Fungsi *autocorrelation* dapat digunakan untuk 2 tujuan yaitu mendeteksi *non-randomness* pada data dan untuk mengidentifikasi *time series model* jika data tidak random. *Autocorrelation* merupakan koefisian korelasi namun dibanding melakukan korealasi pada 2 variabel berbeda, korelasinya berada diantara 2 nilai pada satu variabel pada waktu Xi dan Xi+k[11]. Pada penelitian ini, *autocorrelation* (AR) dihitung menggunakan persamaan:

$$AR[k] = \frac{\sum_{n=k}^{N-(k+1)}(D[n] - E\,D[n])(D[n+k] - E\,D[n])}{\sum_{n=0}^{N-1}(D[n] - E\,D[n])^2} \tag{2.11}$$

2.4 *Naive Bayes Classifier* (NBC)

Naive bayes adalah teknik prediksi berbasis probabilistik sederhana yang berdasar pada penerapan teorema Bayes [12]

1. Asumsi independensi (ketidaktergantungan) yang kuat.
2. Model yang digunakan adalah model fitur independen. Independensi yang kuat pada fitur adalah bahwa sebuah fitur pada sebuah data tidak ada kaitannya dengan adanya atau tidak adanya fitur yang lain dalam data yang sama.

Contoh: kasus klasifikasi hewan dengan fitur: penutup kulit, melahirkan, berat, dan menyusui

1. Dalam dunia nyata, hewan yang berkembang biak dengan cara melahirkan dipastikan hewan tersebut menyusui juga, disini ada ketergantungan pada fitur menyusui karena hewan yang menyusui biasanya melahirkan, atau hewan yang bertelur biasanya tidak menyusui.
2. Dalam Bayes, hal tersebut tidak dipandang, sehingga masing- masing fitur seolah tidak ada hubungan apa-apa.

Kasus lain: prediksi hujan

1. Hujan tergantung angin, cuaca kemarin, kelembaba udara (tidak ada kaitan satu sama lain)
2. Tapi juga tidak boleh memasukkan fitur lain yang tidak ada hubungannya dengan hujan, seperti: gempa bumi, kebakaran, dsb.

Naïve bayes adalah suatu metode pengklasifikasian paling sederhana dengan menggunakan peluang yang ada, dimana diasumsikan bahwa setiap variable X bersifat bebas (independence). Karena asumsi variabel tidak saling terikat, maka didapatkan persamaan (2.12).

$$P(X \mid Y = y) = \prod_{i=1}^{q} P(X_i \mid Y = y) \quad (2.12)$$

Data yang digunakan dapat bersifat kategorial (diskrit) maupun kontinyu. Namun, pada tugas akhir ini akan digunakan data kontinyu, karena hasil ekstraksi fitur merupakan data kontinyu berupa angka-angka hasil perhitungan *mean-centroid, std-centroid, mean-rolloff, std-rolloff, mean-flux, std-flux, mean-zerocrossings, std-zerocrossings,lowenergy*. Maka dari itu Untuk data kontinyu dapat diselesaikan dengan menggunakan langkah-langkah berikut.

Training :

1. Hitung rata-rata (mean) tiap fitur dalam dataset training dengan persamaan (2.13).

$$\mu = \frac{\sum x_i}{n} \quad (2.13)$$

Dimana:

μ = mean

n = banyaknya data

$\sum xi$ = jumlah nilai data

2. Kemudian hitung nilai varian dari diat dataset training tersebut seperti pada persamaan (2.14).

$$\sigma 2 = \frac{1}{(n-1)} \sum_{i=1}^{n} (x_i - \mu)^2 \quad (2.14)$$

Dimana:

$\sigma 2$= varians

μ= mean

xi = nilai data

n = banyaknya data

Testing :

1. Hitung probabilitas (Prior) tiap kelas yang ada dengan cara menghitung jumlah data tiap kelas dibagi jumlah total data secara keseluruhan.

2. Selanjutnya menghitung densitas probabilitasnya menggunakan persamaan (10). Fungsi densitas mengekspresikan probabilitas relatif. Data dengan mean μ dan standar deviasi σ, fungsi densitas probabilitasnya adalah :

$$\varphi_{\mu\sigma}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (2.15)$$

Dimana :

x = data masukan

π = 3,14

σ = standar deviasi

σ^2 = varian

μ = mean

3. Setelah didapatkan nilai densitas probabilitasnya, selanjutnya menghitung posterior masing-masing kelas dengan menggunakan persamaan (2.16).

$$Posterior = \frac{P(Ci) \text{x } P(X1 \ldots\ldots\ldots\ldots\ldots\ldots Xn|Ci)}{P(X1 \ldots\ldots\ldots\ldots\ldots\ldots Xn|Ci)}$$

atau bisa ditulis

$$posterior = \frac{\text{prior x likehood}}{\text{evidence}} \quad (2.16)$$

4. Nilai posterior terbesar adalah kelas yang sesuai.

2.5 Perhitungan akurasi sistem

Setelah proses klasifikasi selesai dilakukan dan didapatkan kelasnya, maka akan dihitung akurasinya dengan persamaan:

$$\text{Akurasi} = \frac{\text{Jumlah data testing yang tepat klasifikasinya}}{\text{Jumlah data testing}} \text{ x } 100\% \qquad (2.17)$$

2.6 Pemrograman Berorientasi Objek

Metodologi beorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek [13].

Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut:

1. Meningkatkan produktivitas

   Kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk massalah lainnya yang melibatkan objek tersebut (*reusable*).
2. Kecepatan pengembangan

   Sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengkodean.
3. Kemudahan pemeliharaan

   Dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering berubah-ubah.
4. Adanya konsistensi

   Sifat pewaris dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.
5. Meningkatkan kualitas perangkat lunak

Pendekatan pengembangan lebih dekat dengan dunia nyata dan adanya konsistensi pada saat pengembangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

Berikut ini adalah beberapa konsep dasar metodologi berorientasi objek:

1. Kelas (*class*)

   Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Sebuah kelas akan mempunyai sifat (atribut), kelakuan (operasi/metode), hubungan (*relationship*) dan arti. Suatu kelas dapat diturunkan dan kelas yang lain, dimana atribut dan kelas semula dapat diwariskan ke kelas yang baru. Secara teknis, kelas adalah sebuah struktur tertentu dalam pembuatan perangkat lunak. Kelas merupakan bentuk struktur pada kode program yang menggunakan metodologi berorientasi objek. Kelas secara fisik adalah berkas atau file yang berisi kode program, di mana kode program merupakan semua hal yang terkait dengan nama kelas.

2. Objek (*object*)

   Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Secara teknis, sebuah kelas saat program dieksekusi maka akan dibuat sebuah objek. Objek dilihat dari segi teknis adalah elemen pada saat *runtime* yang akan diciptakan, dimanipulasi, dan dihancurkan saat eksekusi sehingga sebuah objek hanya ada saat sebuah program dieksekusi.

3. Metode (*method*)

   Metode pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek. Metode dapat berasal dari *event*,

aktivitas atau aksi keadaan, fungsi, atau kelakuan dunia nyata. Contoh metode misalnya *read*, *write*, *move*, *copy*.

4. Atribut (*attribute*)

   Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut dipunyai secara individual oleh sebuah objek, misalnya berat, jenis, nama, dan sebagainya. Atribut sebaiknya bersifat privat untuk menjaga konsep enkapsulasi.

5. Abstraksi (abstraction)

   Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

6. Enkapsulasi (*encapsulation*)

   Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek yang lain mengetahui cara kerjanya.

7. Pewaris (*inheritance*)

   Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dan dirinya.

8. Antarmuka (*interface*)

   Antarmuka atau interface sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi. Deklarasi metode pada sebuah *interface* dapat diimplementasikan oleh kelas lain. Sebuah kelas dapat mengimplementasikan lebih dari satu antarmuka dimana kelas ini akan mendeklarasikan metode pada antarmuka yang dibutuhkan oleh kelas itu. Metode pada antarmuka yang diimplementasikan pada suatu kelas harus sama persis dengan yang ada pada antarmuka. Antarmuka atau *interface* biasanya digunakan agar kelas yang lain tidak mengakses langsung ke suatu kelas, mengakses antarmukanya.

2.7 *Unified Modelling Language* (UML)

UML adalah suatu bahasa yang digunakan untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan suatu sistem informasi. UML dikembangkan sebagai suatu alat untuk analisis dan desain berorientasi objek oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson [14].

2.7.1 *Use Case Diagram*

Use case diagram merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. Diagram ini juga mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. *Use case*, digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa yang berhak menggunakan fungsi-fungsi tersebut.

Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu:

1. Aktor

   Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri. Simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.

2. *Use case*

   Use case merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

2.7.2 *Activity Diagram*

Activity diagram merupakan diagram menggambarkan *workflow* (alur kerja) atau aktifitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak.

2.7.3 *Sequence Diagram*

Sequence diagram merupakan diagram yang menggambarkan kelakuan perilaku objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan (*message*) yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram *sekuen* maka harus diketahui objek-objek yang terlibat

dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang *diinstansiasi* menjadi objek itu.

2.7.4 *Class Diagram*

Class diagram atau diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode. Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak dapat sinkron.

Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

1. Kelas main

   Kelas main merupakan kelas yang mempunyai fungsi awal yang dieksekusi ketika sistem dijalankan.
2. Kelas yang menangani tampilan sistem (*view*)

   Kelas ini mendefinisikan dan mengatur tampilan ke pengguna.
3. Kelas yang diambil dari pendefinisian *use case* (*controller*)

   Kelas ini menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case*. Kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.
4. Kelas yang diambil dari pendefinisian data (model).

   Kelas ini digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data. Semua tabel yang dibuat di basis data dapat dijadikan kelas.

2.8 Perangkat Lunak Pendukung

Perangkat lunak pendukung merupakan perangkat lunak yang digunakan untuk membantu dalam menyelesaikan penelitian.

2.8.1 JAVA

JAVA adalah bahasa pemrograman tingkat tinggi yang berorientasi objek dan program java tersusun dari bagian yang disebut kelas. Kelas terdiri atas metode-

metode yang melakukan pekerjaan dan mengembalikan informasi setelah melakukan tugasnya. Para pemrogram Java banyak mengambil keuntungan dari kumpulan kelas di pustaka kelas Java, yang disebut dengan *Java Application Programming Interface* (API). Kelas-kelas ini diorganisasikan menjadi sekelompok yang disebut paket *(package).* Java API telah menyediakan fungsionalitas yang memadai untuk menciptakan *applet* dan aplikasi canggih. Jadi ada dua hal yang harus dipelajari dalam Java, yaitu mempelajari bahasa Java dan bagaimana mempergunakan kelas pada Java API. Kelas merupakan satu-satunya cara menyatakan bagian eksekusi program, tidak ada cara lain. Pada Java program javac untuk mengkompilasi file kode sumber Java menjadi kelas-kelas *bytecode*. File kode sumber mempunyai ekstensi *.java. Kompilator javac menghasilkan file *bytecode* kelas dengan ekstensi *.class. Interpreter merupakan modul utama sistem Java yang digunakan aplikasi Java dan menjalankan program *bytecode* Java.

Beberapa keunggulan java yaitu java merupakan bahasa yang sederhana. Java dirancang agar mudah dipelajari dan digunakan secara efektif. Java tidak menyediakan fitur-fitur rumit bahasa pemrograman tingkat tinggi, serta banyak pekerjaan pemrograman yang mulanya harus dilakukan manual, sekarang digantikan dikerjakan Java secara otomatis seperti dealokasi memori. Bagi pemrogram yang sudah mengenal bahasa C++ akan cepat belajar susunan bahasa Java namun harus waspada karena mungkin Java mengambil arah (semantiks) yang berbeda dibanding C++. [15]

2.8.2 NetBeans IDE 8.2

NetBeans merupakan sebuah aplikasi *Integrated Development Environment* (IDE) yang berbasiskan *Java* dari Sun Microsystems yang berjalan diatas *Swing. Swing* merupakan sebuah teknologi *java* untuk pengembangan aplikasi *desktop* yang dapat berjalan pada berbagai macam *plaform* seperti *windows, linux, Mac OS X,* dan *Solaris.* Sebuah IDE merupakan lingkup pemrograman yang diintegrasikan kedalam suatu aplikasi perangkat lunak yang menyediakan *Graphic*

User Interface (GUI), suatu kode editor atau *text* , suatu *compiler* dan suatu *debugger.*

Netbeans juga digunakan oleh sang *programmer* untuk menulis, meng-*compile,* mencari kesalahan dan menyebarkan program netbeands yang ditulis dalam bahasa pemrograman lainnya dan program ini pun bebas untuk digunakan dan untuk membuat *professional desktop, enterprise, web, and mobile applications* dengan *Java language, C/C++,* dan bahakan *dynamic languages* seperti PHP, *JavaScript, Groovy,* dan *Ruby* [16].

2.8.3 Microsoft Visio

Microsoft Visio (atau sering disebut Visio) adalah sebuah program aplikasi komputer yang sering digunakan untuk membuat diagram, diagram alir (*flowchart*), *brainstorm*, dan skema jaringan yang dirilis oleh Microsoft Corporation. Aplikasi ini menggunakan grafik vektor untuk membuat diagram-diagramnya.

Visio aslinya bukanlah buatan Microsoft Corporation, melainkan buatan Visio Corporation, yang diakusisisi oleh Microsoft pada tahun 2000. Versi yang telah menggunakan nama Microsoft Visio adalah Visio 2002, Visio 2003, Visio 2007, Visio 2010, Visio 2013, dan Visio 2016 yang merupakan versi terbaru. Visio 2013 Standard dan Professional menawarkan antarmuka pengguna yang sama, tapi seri Professional menawarkan lebih banyak pilihan template untuk pembuatan diagram yang lebih lanjut dan juga penataan letak (layout). Selain itu, edisi Professional juga memudahkan pengguna untuk mengoneksikan diagram-diagram buatan mereka terhadap beberapa sumber data dan juga menampilkan informasi secara visual dengan menggunakan grafik.

2.9 Pengujian *Black Box*

Pengujian *black box*, dapat juga disebut pengujian perilaku, berfokus pada persyaratan fungsional perangkat lunak. Artinya, teknik pengujian *black box* memungkinkan Anda memperoleh serangkaian kondisi masukan yang akan sepenuhnya menjalankan semua persyaratan fungsional untuk sebuah program. Pengujian *black box* bukanlah alternatif teknik *white-box*. Sebaliknya, ini adalah pendekatan pelengkap yang cenderung menemukan kelas kesalahan yang berbeda dari pada metode *whitebox*. Pengujian *black box* mencoba menemukan kesalahan dalam kategori berikut: (1) fungsi yang salah atau hilang, (2) kesalahan antarmuka, (3) kesalahan dalam struktur data atau akses database eksternal, (4) kesalahan perilaku atau kinerja, dan (5) Inisialisasi dan kesalahan terminasi[17].

2.10 K-*Fold Cross Validation*

Cross Validation merupakan salah satu teknik untuk menilai/memvalidasi keakuratan sebuah model yang dibangun berdasarkan dataset tertentu. Pembuatan model biasanya bertujuan untuk melakukan prediksi maupun klasifikasi terhadap suatu data baru yang boleh jadi belum pernah muncul di dalam dataset. Data yang digunakan dalam proses pembangunan model disebut data latih/training, sedangkan data yang akan digunakan untuk memvalidasi model disebut sebagai data test. Salah satu metode cross-validation yang populer adalah K-*Fold Cross Validation*. Dalam teknik ini dataset dibagi menjadi sejumlah K-buah partisi secara acak. Kemudian dilakukan sejumlah K-kali eksperimen, dimana masing-masing eksperimen menggunakan data partisi ke-K sebagai data testing dan memanfaatkan sisa partisi lainnya sebagai data training. Sebagai gambaran, jika melakukan 5-*Fold Cross-Validation* maka desain data eksperimennya sebagai berikut:

Dataset:

| K1 | K2 | K3 | K4 | K5 |

Data Eksperimen:

Eksperimen Ke	Data Latih	Data Test
1	K2,K3,K4,K5	K1
2	K1,K3,K4,K5	K2
3	K1,K2,K4,K5	K3
4	K1,K2,K3,K5	K4
5	K1,K2,K3,K4	K5

Untuk mendapatkan nilai akurasi ataupun ukuran penilaian lainnya dari hasil eksperimen yang kita lakukan, dapat diambil nilai rataan dari seluruh eksperimen tersebut[18].

2.11 *Confusion Matrix*

Confusion Matrix merupakan sebuah matriks yang menunjukkan hasil prediksi dan klasifikasi sebenarnya. Sebuah *confusion matrix* dengan ukuran l x l, dimana l adalah jumlah nilai label yang berbeda. Contohnya ditunjukkan pada gambar 2.4[19].

↓ actual \ predicted→	negative	positive
negative	a	b
positive	c	d

Gambar 2. 4 Confusion Matrix

Dimana untuk menghitung akurasi (a+d)/(a+b+c+d).