# BAB 2
# TINJAUAN PUSTAKA

## 2.1 Optical Character Recognition

OCR adalah singkatan dari *Optical Character Recognition*. Teknologi ini memungkinkan untuk mengenali karakter secara otomatis melalui mekanisme optik. Dalam kasus manusia, mata manusia adalah mekanisme optik. Gambar yang dilihat oleh mata adalah input untuk otak. Kemampuan untuk memahami input ini bervariasi pada setiap orang menurut banyak faktor. OCR adalah teknologi yang berfungsi seperti kemampuan membaca manusia. Meskipun OCR tidak mampu bersaing dengan kemampuan membaca manusia [8].

OCR adalah teknologi yang memungkinkan mengonversi berbagai jenis citra seperti citra kertas yang dipindai, file PDF atau gambar yang diambil oleh kamera digital menjadi data yang dapat diedit dan dicari. Gambar yang diambil oleh kamera digital berbeda dari citra atau gambar yang dipindai.

Secara umum proses OCR dapat dilihat pada Gambar 2.1, dengan penjelasan sebagai berikut[9]:

a. *File Input*

   *File input* berupa *file* citra digital dengan format *.bmp atau *.jpg.

b. *Preprocessing*

   *Preprocessing* adalah proses yang bertujuan untuk menghilangkan bagian-bagian yang tidak diperlukan padagambar *input* untuk proses selanjutnya

c. Segmentasi

   Segmentasi adalah proses membagi daerah yang ingin diamati(*region*) pada tiap karakter yang dideteksi.

d. Normalisasi

   Normalisasi adalah proses merubah ukuran *region* tiap karakter dan ketebalan karakter.

e. Ekstraksi ciri

Ekstraksi ciri adalah proses untuk mengambil ciri-ciri tertentu dari karakter yang diamati.

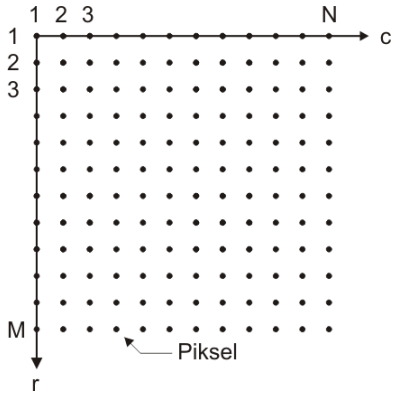
f. *Recognition*

*Recognition* adalah proses pengenalan karakter yang dilakukan dengan cara membandingkan ciri-ciri karakter yang diperoleh dengan ciri-ciri karakter yang ada pada basis data


**Gambar 2.1 Proses OCR Secara Umum**

## 2.2 Citra

Citra adalah gambar dua dimensi yang dihasilkan dari gambar analog dua dimensi yang kontinu menjadi gambar diskrit melalui proses sampling. Citra dilambangkan dengan fungsi dua dimensi, f(x,y) dimana x dan y adalah koordinat spasial. Citra berwarna terdiri dari beberapa citra monokrom, misalnya RGB terdiri dari 3 kanal dan terbentuk dari kombinasi tiga warna yaitu, merah, hijau dan bir Citra digital digambarkan dalam bentuk matriks dimana indeks baris dan kolomnya menyatakan suatu titik pada citra yang disebur piksel dan elemen matriksnya menyatakan tingkat keabuan pada titik tersebut [10]. Persilangan antara baris dan kolom tertentu disebut dengan piksel. Gambar/titik diskrit pada baris (x) dan kolom (y) adalah piksel [x,y].




**Gambar 2.2 Koordinat Piksel**

Citra dapat ditulis dalam bentuk matriks berikut [11]:

$$f(x) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \vdots & f(M-1,N-1) \end{bmatrix} \quad \textbf{(2.1)}$$

## 2.3 Pengolahan Citra

Pengolahan citra adalah pemrosesan sebuah gambar dua dimensi secara digital, proses – proses yang dapat dilakukan dalam pengolahan citra digital adalah sebagai berikut [11]:

1. Merubah dari citra berwarna ke dalam citra abu-abu.
2. Merubah dari citra berwarna atau citra abu-abu ke dalam citra biner. Proses ini dapat dilakukan dengan deteksi tepi atau filtering.

Tujuan dari pengolahan citra digital adalah untuk memperbaiki kualitas suatu gambar agar dapat dengan mudah dipahami oleh mata manusia dan komputer untuk mengolah informasi yang terdapat pada gambar tersebut untuk keperluan pengenalan objek secara otomatis.

## 2.4 Sertifikat

Menurut Kamus Besar Bahasa Indonesia (KBBI) arti dari kata sertifikat merupakan tanda atau surat keterangan (pernyataan) tertulis atau tercetak dari orang

yang berwenang yang dapat digunakan sebagai bukti kepemilikan atau suatu kejadian [12].

Sertifikat memiliki fungsi untuk membuktikan bahwa seseorang telah mengikuti suatu kegiatan *training*, kegiatan workshop, kegiatan seminar, kegiatan kemah pramuka dan kegiatan lainnya yang dilaksanakan dengan maksud memberikan edukasi kepada pesertanya. Terdapat beberapa elemen penting yang biasanya dibutuhkan di dalam sebuah sertifikat Bagian-bagian sertifikat tersebut adalah sebagai berikut [31] :

1. Pemilik Sertifikat
2. Nama Acara Seminar
3. Waktu dan Tempat
4. Pemilik Nama Penyelenggara Acara

Dan ini bisa dilihat pada **Gambar 2.3** pada berikut ini **:**


**Gambar 2.3 Contoh Sertifikat** [31]

## 2.5 Preprocessing

Perancangan sistem *preprocessing* menjelaskan alur kerja dalam OCR dalam menyiapkan citra sebagai data input yang siap diolah lebih lanjut oleh sistem. Sistem preprocessing yang digunakan pada penelitian ini adalah sebagai berikut:

### 2.5.1 Grayscale

Citra *grayscale* adalah citra yang hanya memiliki 1 buah kanal sehingga yang ditampilkan hanyalah nilai intensitas atau dikenal juga dengan istilah derajat keabuan. Karena jenis citra ini hanya memiliki 1 kanal saja, maka citra *grayscale* memiliki tempat penyimpanan yang lebih hemat. Jenis ini disebut juga sebagai 8-bit *image* karena untuk setiap nilai pikselnya memerlukan penyimpanan sebesar 8-bit [13]. Rumus yang digunakan untuk mengubah citra RGB menjadi citra *grayscale* yaitu "luma" atau "*luminance*". Berikut adalah rumus (2.2) untuk mengubah citra menjadi *grayscale*:

$$\boldsymbol{Gray = 0.299 * red + 0.587 * green + 0.114 * blue} \quad \textbf{(2.2)}$$

### 2.5.2 Thresholding

Citra biner atau citra hitam putih (black and white image) adalah citra yang hanya memiliki 2 kemungkinan nilai untuk setiap pikselnya, yaitu 0 atau 1. Nilai 0 akan tampil sebagai warna hitam sedangkan nilai 1 akan tampil sebagai warna putih. Maka dari itu, jenis citra ini hanya membutuhkan 1-bit untuk menyimpan setiap nilai pada setiap pikselnya. Jenis citra ini sering digunakan untuk proses masking ataupun proses segmentasi citra [13]. Proses threshodling digunakan untuk mengekstrak foreground (tinta) dari background (kertas) dan mengubah menjadi citra biner. Proses thresholding mengubah warna gambar menjadi citra biner (binary image) dimana ditentukan sebuah nilai level threshold kemudian piksel yang memiliki nilai level dibawah level threshold di set menjadi warna putih (1 pada nilai biner) dan nilai diatas nilai threshold di set menjadi warna hitam (1 pada nilai biner)[14]. Berikut adalah rumus (2.3) biner yang digunakan:

$$g(x,y) = \begin{cases} 1 \; if \; f(x,y) < T \\ 0 \; if \; f(x,y) > T \end{cases} \quad \textbf{(2.3)}$$

Keterangan:

*Gray* merupakan nilai hasil *grayscale* dan 128 adalah nilai level threshold.

## 2.6 Segmentasi

Segmentasi adalah sebuah proses yang bertujuan untuk membagi sebuah citra menjadi daerah pilihan atau mengisolasi objek dari citra secara keseluruhan, segmentasi dapat dilakukan berdasarkan tekstur, kecerahan, serta intentitas jumlah piksel. Pada proses ini akan melakukan pembagian citra menjadi dua wilayah, yaitu wilayah latar dan wilayah teks. Setelah melakukan pembagian, OCR akan melakukan proses selanjutnya hanya pada wilayah teks yang sudah terbagi[15]. Pada penelitian ini metode segmentasi yang digunakan adalah profil proyeksi, metode ini bekerja dengan dua tahap yaitu mencari baris dari citra (*horizontal)* dan mencari karakter (*vertical)*. Profil Proyeksi akan mencari garis pada teks secara vertikan dan horizontal, dimana proyeksi horizontal akan mendapatkan baris teks kemudian proyeksi vertikal akan memisahkan setiap kolom karakter. Dimana S(N, M) merupakan citra biner dengan N baris dan M kolom. Profil proyeksi terdapat 2 jenis yaitu[16]:

1. Profil Vertikal

   Menjumlahkan pixel putih yang tegak lurus dengan sumbu y, yang diwakili vektor $P_{ver}$ dengan ukuran N yang didefinisikan sebagai:

$$P\ ver\ [k] = \sum_{j=1}^{m} I[bk, k] \qquad \textbf{(2.4)}$$

   Dengan ketentuan sebagai berikut :

   M = tinggi citra

   Pver[k] = jumlah piksel pada kolom k citra

2. Profil Horizontal

   Menjumlahkan pixel putih yang tegak lurus dengan sumbu x, yang diwakili vektor Phor dengan ukuran M yang didefinisikan sebagai:

$$P\ hor\ [b] = \sum_{j=1}^{n} I[b, kj] \qquad \textbf{(2.5)}$$

   Dengan ketentuan sebagai berikut :

N = lebar citra

Phor[b] = jumlah piksel pada baris b citra

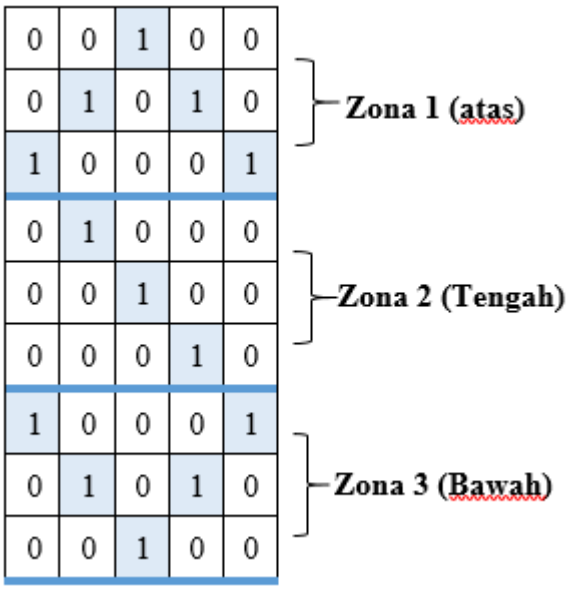
## 2.7 Ekstraksi Ciri

Ekstraksi ciri adalah proses pengukuran terhadap data yang telah di normalisasi untuk membentuk sebuah nilai fitur. Nilai fitur digunakan oleh pengklasifikasi untuk mengenali unit masukan dengan unit target keluaran dan memudahkan pengklasifikasian karena nilai ini mudah untuk dibedakan[17].

Ekstraksi ciri (feature extraction) dapat dikatakan sebagai proses fundamental dari sistem pengenalan karakter. Fitur/ciri adalah karakteristik unik dari suatu objek [18]. Tujuan dari ekstraksi ciri ini adalah mendapatkan karakteristik suatu karakter yang berguna untuk membedakan antara karakter yang satu dengan lainnya. Ciri yang baik adalah ciri yang memiliki daya pembeda tinggi, sehingga proses klasifikasi pada ciri karakter bisa mendapatkan akurasi yang baik. Pada penelitian ini, metode yang digunakan untuk ekstraksi ciri adalah metode *zoning*.

### 2.7.1 *Zone Based Feature Extraction*

*Zoning* adalah salah satu ekstraksi fitur yang paling popular dan sederhana untuk diimplementasikan [19]. Setiap citra dibagi menjadi N x M zona dan dari setiap zona tersebut dihitung nilai fitur sehingga didapatkan fitur dengan panjang N x M. Salah satu cara menghitung nilai fitur setiap zona adalah dengan menghitung jumlah piksel hitam setiap zona dan membaginya dengan jumlah piksel hitam terbanyak pada yang terdapat pada salah satu zona. Contoh pembagian 3 zona pada citra biner dapat dilihat pada Gambar 2.4 [19].




**Gambar 2.4 Pembagian Zona pada Citra Biner**

Metode ekstraksi fitur berbasis zona memberikan hasil yang baik bahkan ketika langkah sebelum proses tertentu dimulai seperti *filtering*, *smoothing*, dan menghapus zona yang tidak dianggap. Konsep metode ekstraksi ciri yang digunakan untuk mengekstraksi fitur untuk klasifikasi yang efisien dan pengenalan yaitu [19]:

1. Hitung *centroid* dari citra. Menghitung centroid dari citra biner masukan dengan persamaan **(2.6)** dan **(2.7)**.

   Rumus mencari *centroid* X

$$C_x = \frac{(x_1.p_1 + x_2.p_2 + \cdots + x_n.p_n)}{(p_1 + p_2 + \cdots + p_n)} \quad \textbf{(2.6)}$$

   Rumus mencari *centroid* Y

$$C_y = \frac{(y_1.p_1 + y_2.p_2 + \cdots + y_n.p_n)}{(p_1 + p_2 + \cdots + p_n)} \quad \textbf{(2.7)}$$

   Keterangan :

a. Cx = *centroid* koordinat x
b. Cy = *centroid* koordinat y
c. Xn = koordinat x dari piksel ke-n
d. Yn = koordinat y dari piksel ke-n
e. Pn = nilai piksel ke-n

2. Bagi matriks kedalam *n* buah zona yang sama besar proporsinya.
3. Hitung jarak antara titik *centroid* dengan koordinat *pixel* yang memiliki nilai. Persamaan untuk menghitung jarak piksel.

$$d(P,C) = \sqrt{(x_p - C_x)^2 + (y_p - C_y)^2} \quad \textbf{(2.8)}$$

Keterangan :

a. d = jarak antara dua titik
b. P = koordinat piksel
c. C = koordinat centroid
d. $X_p$ = koordinat piksel X
e. $Y_p$ = koordinat piksel Y
f. $C_x$ = koordinat centroid X
g. $C_y$ = koordinat centroid Y

4. Ulangi langkah 3 untuk *pixel* yang ada di semua zona.
5. Hitung rata-rata dari jarak yang telah didapat pada langkah 3.

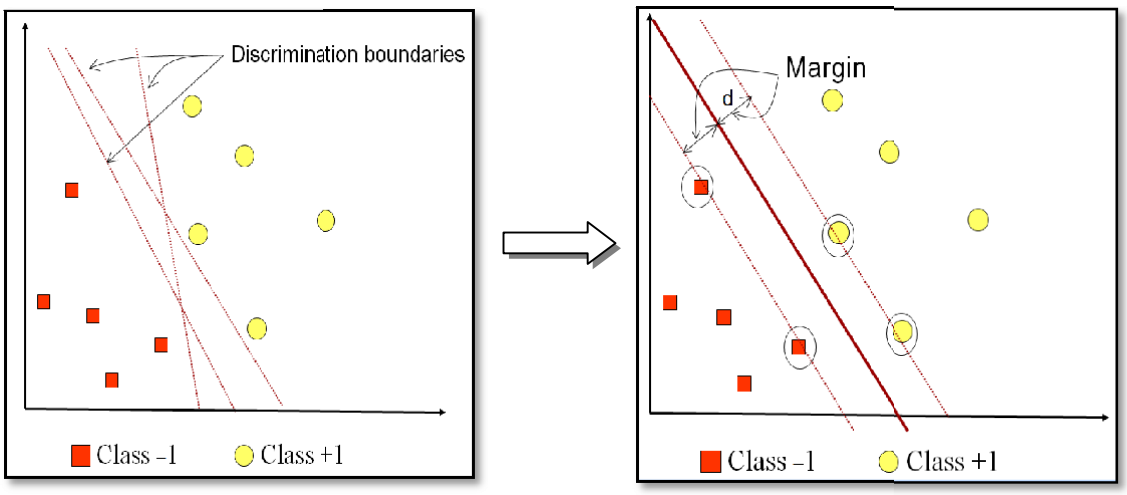
$$rerata\ jarak = \sum \frac{d(P,C)}{\sum P} \quad \textbf{(2.9)}$$

6. Ulangi langkah 5 hingga didapat masing-masing rata-rata jarak dari setiap zona.
7. Akhirnya *n* buah fitur akan didapat untuk melakukan klasifikasi dan pengenalan

## 2.8 Support Vector Machine

*Support Vector Machine* (SVM) pertama kali diperkenalkan oleh Valdimir Vanpik pada tahun 1992 sebagai rangkaian harmonisasi konsep-konsep unggulan dalam bidang pengenalan pola [20]. Support Vector Machine merupakan metode

pembelajaran yang digunakan untuk klasifikasi biner, ide dasarnya adalah mencari hyperplane terbaik yang berfungsi sebagai pemisah dua class pada input space[20]. Berikut adalah ilustrasi dari pemilihan hyperplane terbaik untuk memisahkan dua kelas data :




(a) (b)

**Gambar 2.5 (a) hyperplane yang Kurang Baik/Tepat, (b) Hyperplane Terbaik**

Berbeda dengan pendekatan neural network yang berusaha mencari hyperplane pemisah antar kelas, SVM mencoba mencari dan memisah hyperplane yang terbaik yang berada pada dua kelas, gambar (a) menunjukan pola-pola yang merupakan anggota dari dua buah kelas +1 dan -1, pola yang berada pada anggota -1 kemudian disimbolkan dengan warna merah, sedangkan pola pada +1 warna kuning. Masalah pada klasifikasi dapat dilakukan dengan usaha menemukan garis (hyperplane) yang memisahkan kelompok tersebut. Pendefinisian persamaan suatu hyperplane pemisah yang dituliskan dengan:

$$w * x_i + b = 0 \qquad \textbf{(2.10)}$$

Data $x_i$ yang terbagi dalam dua kelas, yang termasuk kelas -1 (sample negatif) didefinisikan sebagai vektor yang memenuhi pertidaksamaan 2.6. Sedangkan yang

termasuk kelas +1 (sample positif) memenuhi persamaan 2.7. Berikut adalah persamaan 2.6 dan 2.7 dibawah ini:

$$w * x_i + b < 0 \text{ untuk } y_i = -1 \quad \textbf{(2.11)}$$

$$w * x_i + b > 0 \text{ untuk } y_i = +1 \quad \textbf{(2.12)}$$

Ket:

$x_i$ = data input

$y_i$ = label yang diberikan

$w$ = nilai dari bidang normal

$b$ = posisi bidang relatif terhadap pusat koordinat

Parameter w dan b adalah parameter yang akan dicari nilainya, bila label $y_i = -1$, maka pembatas menjadi persamaan (**2.13)**. Apabila label data yi = +1, maka pembatas menjadi persamaan (**2.14**) Berikut adalah persamaan (**2.14**) dan (**2.14**) dibawah ini:

$$w * x_i + b \leq -1 \quad \textbf{(2.13)}$$

$$w * x_i + b \geq +1 \quad \textbf{(2.14)}$$

Margin terbesar dapat dicari dengan cara memaksimalkan jarak antara bidang pembatas kedua kelas dan titik terdekatnya, yaitu $2/|w|$. Hal ini dirumuskan sebagai permasalahan Quadratic Programming (QP) problem yaitu mecari titik minimal persamaan (**2.15**) dengan memperhatikan persamaan (**2.16**) Berikut:

$$min\tau(w) = \frac{1}{2}\|w\|^2 \quad \textbf{(2.15)}$$

$$y_i(w * x_i + b) - 1 \geq 0, (i = 1, \dots, n) \quad \textbf{(2.16)}$$

Permasalahan ini dapat dipecahkan dengan berbagai teknik komputasi. Lebih mudah diselesaikan dengan mengubah persamaan (**2.16**) Kedalam fungsi

lagrangian pada persamaan (**2.17**) dan menyederhanakannya menjadi persamaan (**2.18**) berikut:

$$L(w,b,a) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} a_i \left(y_i\left((w^T x_i + b) - 1\right)\right) \quad \textbf{(2.17)}$$

$$L(w,b,a) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} a_i y_i (w^T x_i + b) + \sum_{i=1}^{n} a_i \quad \textbf{(2.18)}$$

Dimana ai adalah lagrange multiplier yang bernilai nol atau positif (ai > 0). Nilai optimal dari persamaan (**2.17**) dapat dihitung dengan meminimalkan L terhadap w, b, dan a. Dapat dilihat pada persamaan (**2.19**) Sampai (**2.21**) Berikut:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} a_i y_i x_i = 0 \quad \textbf{(2.19)}$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} a_i y_i = 0 \quad \textbf{(2.20)}$$

$$\frac{\partial L}{\partial a} = \sum_{i=1}^{n} a_i y_i (w^T x_i + b) - \sum_{i=1}^{n} a_i = 0 \quad \textbf{(2.21)}$$

Maka masalah lagrange untuk klasifikasi dapat dinyatakan pada persamaan (**2.22**) berikut:

$$Min\ L(w,b,a) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} a_i y_i (w^T x_i + b) - \sum_{i=1}^{n} a_i \quad \textbf{(2.22)}$$

Dengan memperhatikan persamaan (**2.23** dan (**2.24** berikut:

$$w - \sum_{i=1}^{n} a_i y_i x_i = 0 \quad \textbf{(2.23)}$$

$$\sum_{i=1}^{n} a_i y_i = 0 \quad \textbf{(2.24)}$$

Model persamaan (**2.24**) diatas merupakan primal lagrange. Sedangkan dengan memaksimalkan L terhadap ai, persamaannya menjadi persamaan (**2.25**) berikut:

$$Max \sum_{i=1}^{n} a_i - \frac{1}{2} \sum_{i=1,j=1}^{n} a_i a_j y_i {y_j}^T x_i x_j^T \quad \textbf{(2.25)}$$

Dengan memperhatikan persamaan 2.21 berikut:

$$\sum_{i=1}^{n} a_i y_i = 0, a_i \geq 0 (i, j = 1, \dots, n) \quad \textbf{(2.26)}$$

Untuk mencari nilai xi dan yi dapat dilakukan ketika sudah didapatkan nilai tiap karakter dari pembobotan tf-idf dan inisialisasi kelas. Hasil dari pembobotan tf-idf diubah kedalam bentuk format data SVM, sedangkan data kelas menjadi label.

Untuk mendapatkan nilai ai, langkah pertama adalah mengubah setiap abstrak menjadi nilai vektor (support vector) = $\left(\frac{x}{y}\right)$, kemudian nilai vektor dari setiap karakter dimasukan ke persamaan (**2.27**) kernel trick phi berikut:

$$\emptyset \begin{bmatrix} x \\ y \end{bmatrix} = \begin{cases} \sqrt{x_n^2 + y_n^2} > 2 \; maka \; \begin{bmatrix} \sqrt{x_n^2 + y_n^2} - x + |x - y| \\ \sqrt{x_n^2 + y_n^2} - y + |x - y| \end{bmatrix} \\ \sqrt{x_n^2 + y_n^2} \leq 2 \; maka \; \begin{bmatrix} x \\ y \end{bmatrix} \end{cases} \quad \textbf{(2.27)}$$

Nilai x didapatkan dari persamaan (**2.28**) kernel linier untuk x berikut:

$$\sum_{i=1,j=1}^{n} x_i x_j^T , (i, j = 1, \dots, n) \quad \textbf{(2.28)}$$

Nilai y didapatkan dari persamaan (**2.29**) kernel linier untuk y berikut:

$$\sum_{i=1,j=1}^{n} y_i y_j^T , (i, j = 1, \dots, n) \quad \textbf{(2.29)}$$

Untuk mendapatkan jarak tegak lurus yang optimal dengan mempertimbangkan vektor positif, maka hasil perhitungan dari substitusi nilai x

dan nilai y ke persamaan (**2.27**) diberi nilai bias = 1. Kemudian cari parameter ai, dengan terlebih dahulu mencari nilai fungsi setiap abstrak menggunaan persamaan (**2.30**), lalu mencari nilai ai, pada persamaan linear menggunakan persamaan (**2.30**) dengan memperhatikan i,j = 1,...,n berikut:

$$\sum_{i=1,j=1}^{n} a_i S_i^T S_j \quad \textbf{(2.30)}$$

$$\sum_{i=1,j=1}^{n} a_i S_i^T S_j = y_i \quad \textbf{(2.31)}$$

Setelah parameter ai didapatkan kemudian dimasukkan ke persamaan (**2.32** berikut:

$$\hat{\mathrm{W}} = \sum_{i=1}^{n} a_i S_i \quad \textbf{(2.32)}$$

Hasil yang didapatkan menggunakan persamaan (**2.32**), selanjutnya digunakan persamaan (**2.33**), untuk mendapatkan nilai w dan b:

$$y = wx + b \quad \textbf{(2.33)}$$

Sedemikian sehingga didapatkanlah nilai w dan b atau nilai hyperplane untuk mengklasifikasikan kedua kelas.

Sebuah fungsi bisa menjadi fungsi kernel jika memenuhi Teorema mercer, yang menyatakan bahwa matriks kernel yang dihasilkan harus bersifat positive semi definite [21]. Berikut ini adalah beberapa fungsi kernel yang umum digunakan yaitu:

Kernel linear

$$K(x_i, x) = {x_i}^T x \quad \textbf{(2.34)}$$

Polynomial

$$K(x_i, x) = (\gamma . {x_i}^T x + r)^p, \gamma > 0 \quad \textbf{(2.35)}$$

Radial basis Function

$$K(x_i, x) = exp(-\gamma \|x_i - x_2\|^2), \gamma > 0 \quad \textbf{(2.36)}$$

Sigmoid kernel

$$K(x_i, x) = \tanh(\gamma {x_i}^T + r) \quad \textbf{(2.37)}$$

Hyperplane yang menjadi pemisah terbaik dapat ditemukan dengan mengukur margin hyperplane dan mencari titik maksimalnya. Margin adalah jarak antara hyperplane dengan pattern terdekat dari masing-masing kelas. Pattern yang paling dekat tersebut disebut Support Vector. Untuk pembahasan suatu kasus yang dapat dipisahkan secara linier, maka dalam hal ini fungsi pemisah yang dicari adalah fungsi linier. Fungsi tersebut dapat didefinisikan sebagai:

$$g(x) \coloneqq sqn(f(x)) \quad \textbf{(2.38)}$$

dengan

$$f(x) = WT\ X + b \quad \textbf{(2.39)}$$

Masalah klasifikasi ini dapat dirumuskan berikut: kita akan menemukan set parameter (w,b) sehingga f(x) = < w,x> + b = yi, untuk semua i. Dalam teknik ini kita berusaha menemukan fungsi pemisah (classfiier/hyperplane) terbaik diantara fungsi yang tidak terbatas jumlahnya untuk memisahkan dua macam objek. Hyperplane terbaik adalah hyperplane yang terletak ditengah-tengah antara dua set objek dari dua kelas. Mencari hyperplane terbaik ekuivalen dengan memaksimalkan margin atau jarak antara dua set objek dari kelas yang berbeda. Jika WX1 + b = +1 adalah hyperplane pendukung dari kelas +1 (WX1 + b = +1) dan WX2 + b = -1 hyperplane pendukung dari kelas -1 (WX2 + b = -1), margin antara dua kelas dapat dihitung dengan mencari jarak antara kedua hyperplane-hyperplane pendukung dari kedua kelas. Secara spesifik margin dihitung dengan cara berikut:

$$(WX1 + b = +1) – (WX2 + b = -1) = W(X1 – X2) =2 \quad \textbf{(2.40)}$$

= >

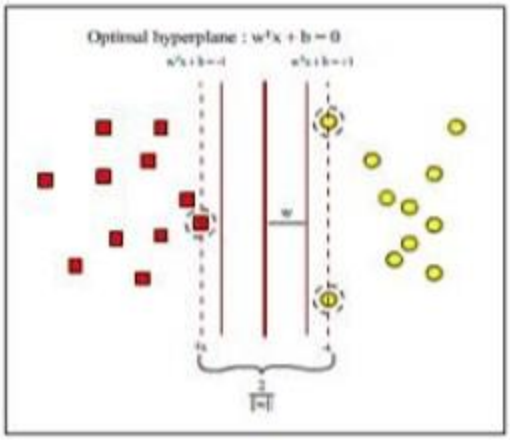
$$LD = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i,j} a_i a_j y_i y_j x_i x_j \quad \textbf{(2.41)}$$

$$syarat\ 1: \sum_{i=1}^{n} a_i y_i = 0$$

$$syarat\ 2: a_i \geq 0, i = 1,2, \dots, N$$

$x_i x_j$ merupakan dot product dua data dalam data latih. Hyperplane (batas keputusan atau pemisah).

Gambar 2.6 menunjukan bagaimana SVM bekerja untuk menentukan suatu fungsi pemisah dengan margin yang maksimal [20].




**Gambar 2.6 Mencari Fungsi Pemisah Yang Optimal Untuk Objek Yang Dapat Dipisahkan Secara Linier**

Untuk membuktikan bahwa memaksimalkan margin antara dua set objek akan meningkatkan probabilitas pengelompokkan secara benar dari data testing. Pada dasarnya jumlah fungsi pemisah tidak terbatas jumlahnya, misalnya dari jumlah yang tak terbatas kita ambil dua fungsi, yaitu f1(x) dan f2(x). Fungsi f1

memiliki margin yang lebih besar daripada f2. Setelah menemukan dua fungsi ini, sekarang suatu data baru masuk dengan keluaran -1. Maka kita harus mengelompokkan apakah data ini ada dalam kelas -1 atau +1 menggunakan fungsi pemisah yang sudah kita temukan. Dengan mengguankan f1, kita akan kelompokkan data baru ini di kelas -1 yang berarti kita benar mengelompokkannya. Kemudian dengan f2 kita akan menempatkannya di kelas +1 yang berarti salah. Dari contoh sederhana ini kita lihat bahwa memperbesar margin bisa meningkatkan probabilitas pengelompokkan suatu data secara benar.

### 2.8.1 Support Vector Machine untuk Multi-Kelas

Pada awal dikembangkannya SVM pendekatan ini digunakan untuk klasifikasi dua kelas. Pengembangan ke arah persoalan klasifikasi untuk multi kelas masih menjadi perhatian peneliti [10].Terdapat dua pendekatan utama kuntuk SVM multi kelas, yang pertama kita dapat menemukan dan menggabungkan beberapa fungsi pemisah persoalan klasifikasi dua kelas untuk menggabungkan beberapa fungsi pemisah persoalan klasifikasi dua kelas untuk menyelesaikan persoalan multi kelas. Kedua, secara langsung menggunakan semua data dari semua kelas dalam satu formulasi persoalan optimasi. Yang termasuk pada pendekatan pertama diama beberapa fungsi untuk problem dua kelas dikembangkan lalu digabung antara lain: satu-lawan-semua (One-Against-All, OAA), dan satu-lawan-satu (One-Against-One, OAO) [20].

### 2.8.2 Metode Satu-Lawan-Semua (One-Against-All)

Dengan menggunakan metode ini, dibangun k sebuah model SVM biner (k adalah jumlah kelas). Setiap model klasifikasi ke-i dilatih dengan menggunakan keseluruhan data, untuk mencari solusi, terdapat permasalahan klasifikasi dengan 4 buah kelas. Untuk *training* digunakan 4 buah SVM biner seperti pada Tabel 2.1 dan penggunaannya dalam mengklasifikasi data baru dapat dilihat pada Gambar .

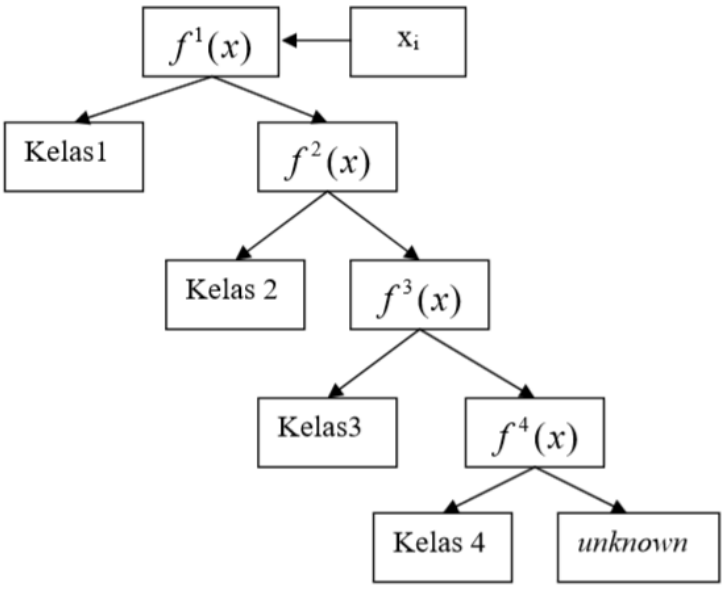
$$\min_{w^i,b^i,\xi^j} \frac{1}{2}\left(w^i\right)^T w^i + C\sum_t \xi_t^i$$
$$\left(w^i\right)^T \emptyset(x_t) + b^i \geq -1 + \xi_t^i \rightarrow y_t \neq i, \quad \textbf{(2.42)}$$
$$\xi_t^i \geq 0$$

**Tabel 2.1 Contoh 4 SVM Biner dengan Metode One Vs All**

| yi = 1 | yi = -1 | Hipotesis |
|---|---|---|
| Kelas 1 | Bukan kelas 1 | $f^1(x) = (w^1)x + b^1$ |
| Kelas 2 | Bukan kelas 2 | $f^2(x) = (w^2)x + b^2$ |
| Kelas 3 | Bukan kelas 3 | $f^3(x) = (w^3)x + b^3$ |
| Kelas 4 | Bukan kelas 4 | $f^4(x) = (w^4)x + b^4$ |

Konsep pada OAA yaitu dimisalkan pada kasus lima kelas, kelas 1, 2, 3, dan 4. Bila akan diujikan $\rho^{(1)}$, semua data dalam kelas 1 diberi label +1 dan data dari kelas lainnya diberi label -1. Pada $\rho^{(2)}$, semua data dalam kelas 2 diberi label +1 dan data dari kelas lainnya diberi label -1 dst hingga data terakhir. Kemudian dicari hyperplane dengan algoritma SVM dua kelas. Maka akan didapat hyperplane untuk masing-masing kelas diatas. Kemudian kelas dari suatu data barus x ditentukan berdasarkan nilai terbesar dari hyperplane [22].

$$kelas\ x = arg\max_{l=1\ldots k}\left(\left(w^{(l)}\right)^T.\emptyset(x) + b^{(l)}\right) \quad \textbf{(2.43)}$$




**Gambar 2.7 Contoh Klasifikasi dengan Metode One Vs All**

## 2.9 Least Square Support Vector Machine

*Least Squares Support Vector Machine* (LS-SVM) merupakan salah satu dari modifikasi SVM konvensional. Jika SVM dikarakteristik oleh permasalahan quadratic programming dengan pembatas berupa pertidaksamaan, maka LS-SVM sebaliknya, diformulasikan dengan menggunakan pembatas berupa persamaan [23]. Sehingga solusi LS-SVM dihasilkan dengan menyelesaikan persamaan linear. Hal yang berbeda dengan penyelesaian SVM yang solusinya didapatkan melalui penyelesaian quadratic programming. Saat ini, LS-SVM banyak dilakukan pada klasifikasi dan estimasi.

Sama seperti algoritma SVM, LS-SVM setelah menentukan nilai *x* yang didapatkan berdasarkan nilai vektor dari ekstraksi ciri, yang kemudian menentukan *y* yang didapatkan dari SVM untuk multi-kelas yaitu metode satu-lawan-semua (*One-Against-All*) yang dimana setiap kelas 1 akan diberikan label + 1 dan untuk kelas data lainnya akan diberi label -1. Setelah itu dilakukannya perhitungan kernel linear yaitu sebagai berikut [21] :

Kernel linear

$$K(x_i, x) = x_i{}^T x \qquad \textbf{(2.44)}$$

Tetapi terdapat perbedaan antara SVM dengan LS-SVM yaitu setelah perhitungan kernel ini, yang dimana apabila SVM setelah menghitung kernel linear ini akan dimasukkan ke fungsi *lagrange multiplier* / LD max [21] :

$$LDmax = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i,j} a_i a_j y_i y_j x_i x_j \quad \textbf{(2.45)}$$

$syarat\ 1$: $\sum_{i=1}^{n} a_i y_i = 0$

$syarat\ 2$: $a_i \geq 0, i = 1,2, \ldots, N$

Setelah perhitungan LD max didapatkan, Dengan demikian, dapat diperoleh nilai $a_i$ yang nantinya digunakan untuk menemukan w. Terdapat nilai $a_i$ untuk setiap data pelatihan. Data pelatihan yang memiliki nilai > 0 $a_i$ adalah support vector sedangkan sisanya memiliki nilai $a_i$ = 0 . Dengan demikian fungsi keputusan yang dihasilkan hanya dipengaruhi oleh support vector.

Formula pencarian bidang pemisah terbaik ini adalah pemasalahan quadratic programming, sehingga nilai maksimum global dari α i selalu dapat ditemukan. Setelah solusi pemasalahan quadratic programming ditemukan (nilai α i ), maka kelas dari data pengujian x dapat ditentukan berdasarkan nilai dari fungsi keputusan [21] :

$$f(x_1) = \sum_{i=1}^{n} a_i y_i K(x_i . x_{uji}) + b \quad \textbf{(2.46)}$$

Berbeda dengan LS-SVM setelah mendapatkan nilai *x* dan *y* beserta didapatkannya nilai kernel linear nya, pehitungan yang digunakan yaitu, misalkan ada n sampel untuk set pelatihan $\{x_k, y_k\}_{k=1}^{n}$ yang dimana, $x_k \in R^n$ adalah masukan, sedangkan $y_k \in R$ adalah keluaran. Di model yang asli, model regresi memiliki bentuk [23] :

$$y(x) = \omega^T \varphi(x) + b \quad \textbf{(2.47)}$$

Dimana, $\omega$ adalah vektor berat dari $\varphi(x)$ dan merupakan fungsi pemetaan nonlinear; $b$ adalah offset. Dan akan membentuk fungsi optimasi sebagai berikut :

$$minJ(\omega, e) = \frac{1}{2}\omega^T\omega + \frac{\gamma}{2}\sum_{k=1}^{n} e_k^2 \qquad \textbf{(2.48)}$$
$$s.t. \rightarrow y_k = \omega^T\varphi(x_k) + b + e_k$$

yang dimana, $\gamma$ adalah faktor dari; $e_k$ adalah sampel point kesalahan. Optimasi permasalahan **(2.45)** diubah menjadi dua ruang, dan *Lagrange function* diperkenalkan:

$$L(\omega, b, e, \alpha) = J(\omega, e) - \sum_{k=1}^{n} a_k[\omega^T\varphi(x_k) + b + e_k - y_k] \qquad \textbf{(2.49)}$$

di mana, *Lagrange multiplier* $a_k$ disebut nilai dukungan.

Persamaan diferensial parsial yang akan dipecahkan untuk setiap variabel sebagai berikut [23]:

$$\begin{Bmatrix} \frac{\partial L}{\partial \omega} = 0, \\ \frac{\partial L}{\partial b} = 0, \\ \frac{\partial L}{\partial \omega} = 0, \\ \frac{\partial L}{\partial \omega} = 0, \end{Bmatrix} \rightarrow \begin{Bmatrix} \omega = \sum_{k=1}^{n} \alpha_k \varphi(x_k) \\ \sum_{k=1}^{n} \alpha_k = 0, \\ a_k = \varphi e_k, \\ \omega^T\varphi(x_k) + b + e_k - y_{k,} \end{Bmatrix} \qquad \textbf{(2.50)}$$

Eliminasi $\omega, e$ dan akan mendapatkan fungsi baru yaitu :

$$\begin{pmatrix} 0 & I_v \\ I_v & \Omega + \frac{1}{\gamma} \end{pmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix} \qquad \textbf{(2.51)}$$

Yang dimana :

$$y = (y_1, y_2, \dots, y_n)^T; I_v = (1,1, \dots, 1)^T; \alpha = (\alpha_1, \alpha_2, \dots, \alpha_n)^T, \qquad \textbf{(2.52)}$$

Maka nilai $\Omega$ Adalah matrix :

$$\Omega_{ij} = \mathrm{y}_k \mathrm{y}_h \varphi(\mathrm{x}_k)^T \varphi(\mathrm{x}_h) = \mathrm{y}_k \mathrm{y}_h \varphi(\mathrm{x}_k, \mathrm{x}_h), h = 1,2, \dots, n. \qquad \textbf{(2.53)}$$

Yang kemudian dihitung nilai $\Omega + \frac{I}{\gamma}$ , kemudiansetelah di dapatkan nilai $\Omega + \frac{I}{\gamma}$ dimasukkan kedalam model :

$$\left[\begin{array}{ccc|c} I & 0 & 0 & -Z^T \\ 0 & 0 & 0 & -Y^T \\ 0 & 0 & \gamma I & -I \\ \hline Z & Y & I & 0 \end{array}\right] \left[\begin{array}{c} w \\ b \\ e \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \\ 0 \\ \hline \vec{1} \end{array}\right] \quad \textbf{(2.51)}$$

Oleh karena itu, model perkiraan LS-SVM regression dapat diperoleh:

$$y(x) = \sum_{k=1}^{n} \alpha_k \varphi(x, x_k) + b \quad \textbf{(2.52)}$$

Dimana $\alpha, b$ solusi dari persamaan **(2.52)** yang dimana nilai *langrange multiplier* bisa bernilai positif ataupun negatif [23], berbeda dengan SVM yang mengharuskan nilai $\alpha$ yaitu positif.

## 2.10 Pemodelan Sistem

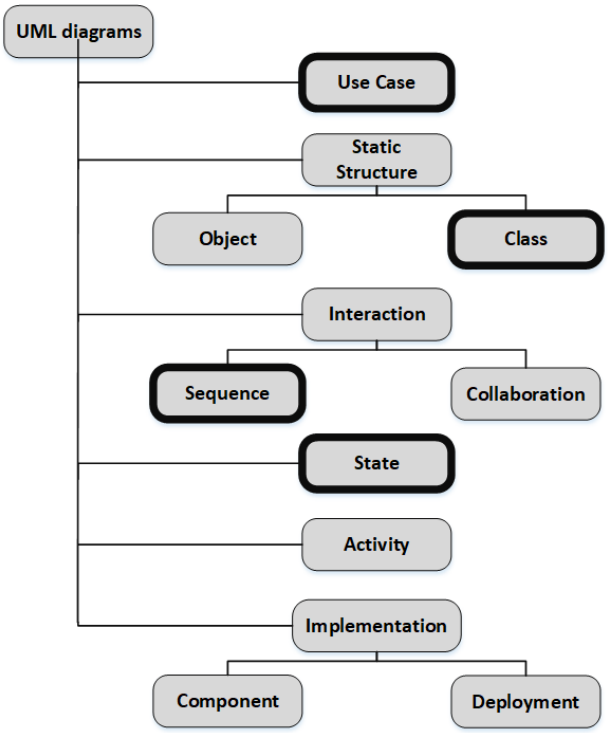
Pada suatu sistem terdapat beberapa proses di dalamnya, yang mana proses tersebut harus dimodelkan untuk memperjelas gambaran yang terjadi pada proses tersebut berjalan. Berikut ini adalah pemodelan sistem yang digunakan pada penelitian ini, yang terdiri dari Blok Diagram, *DFD*, Diagram Konteks, dan *Flowchart.*

## 2.11 UML (Unified Modeling Language)

*Unified Modeling Language* (UML) adalah bahasa pemodelan visual yang digunakan untuk menspesifikasikan, menvisualisasikan, membangun, dan mencitratasikan rancangan dari suatu sistem perangkat lunak [24].

Pemodelan memberikan gambaran yang jelas mengenai sistem yang akan dibangun baik dari sisi struktural ataupun fungsional. UML dapat diterapkan pada semua model pengembangan, tingkatan siklus sistem, dan berbagai macam domain aplikasi. Dalam UML terdapat konsep semantik, notasi, dan panduan masing-masing diagram. UML bertujuan menyatukan teknik-teknik pemodelan berorientasi objek-objek menjadi terstandarisasi[25].

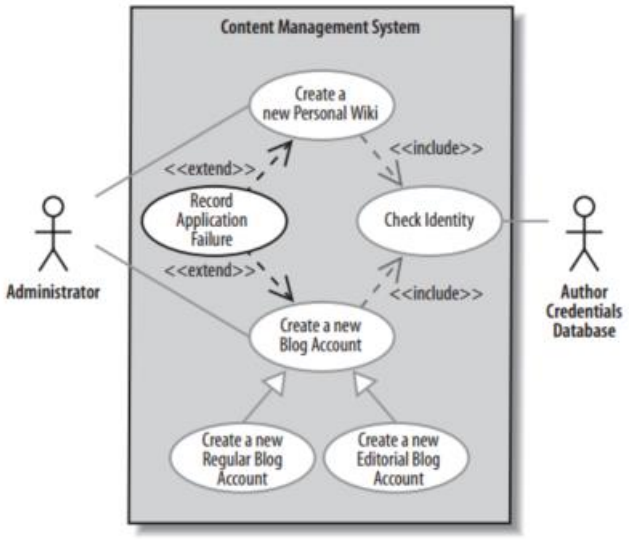
UML dibuat oleh Grady Booch, James Rumbaugh, dan Ivar Jacobson di bawah bendera Rational Software Corps. UML menyediakan notasi-notasi yang membantu memodelkan sistem dari berbagai prespetktif. UML tidak hanya digunakan dalam pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan [25].




**Gambar 2.8 *Unified Modeling Language* Diagram [25].**

### 2.11.1 *Use Case Diagram*

*Use case* diagram menyajikan interakasi antara use case dan aktor. Dimana, aktor dapat berupa orang, peralatan, atau sistem lain yang berinteraksi dengan sistem yang sedang dibangun. Use case menggambarkan fungsionalitas sistem atau persyaratan-persyaratan yang harus dipenuhi sistem dari pandangan pemakai. Sebuah use case digambarkan sebagai elips horizontal dalam suatu diagram UML use case.




**Gambar 2.9 Contoh *Use Case* Diagram [25]**

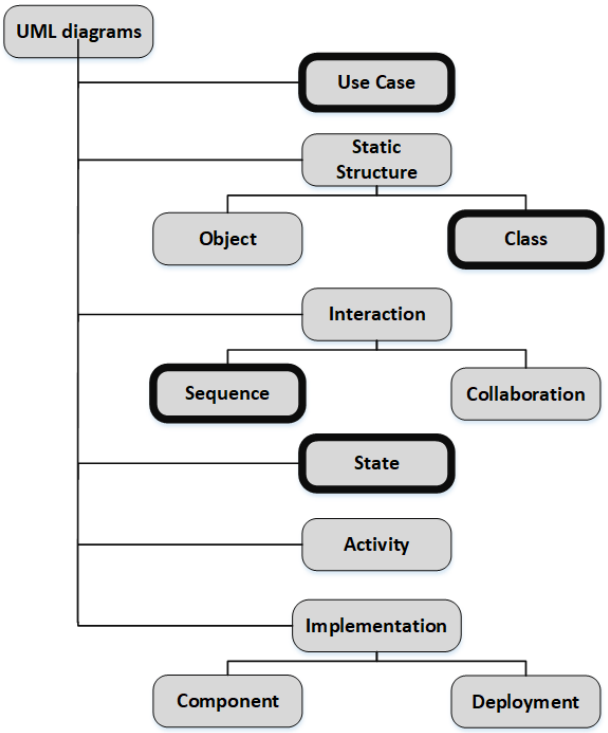
### 2.11.2 *Activity Diagram*

*Activity diagram* merupakan diagram yang menggambarkan workflow (aliran kerja) atau aktivitas dari sebuahUML (*Unified Modeling Language*).

*Unified Modeling Language* (UML) adalah bahasa pemodelan visual yang digunakan untuk menspesifikasikan, menvisualisasikan, membangun, dan mencitratasikan rancangan dari suatu sistem perangkat lunak [24].

Pemodelan memberikan gambaran yang jelas mengenai sistem yang akan dibangun baik dari sisi struktural ataupun fungsional. UML dapat diterapkan pada semua model pengembangan, tingkatan siklus sistem, dan berbagai macam domain aplikasi. Dalam UML terdapat konsep semantik, notasi, dan panduan masing-masing diagram. UML bertujuan menyatukan teknik-teknik pemodelan berorientasi objek-objek menjadi terstandarisasi[25].

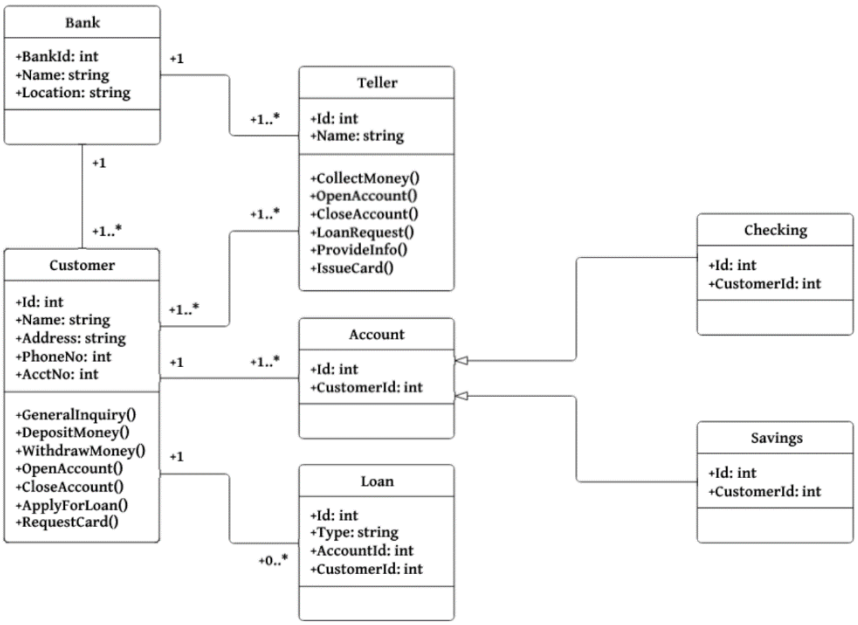
UML dibuat oleh Grady Booch, James Rumbaugh, dan Ivar Jacobson di bawah bendera Rational Software Corps. UML menyediakan notasi-notasi yang membantu memodelkan sistem dari berbagai prespetktif. UML tidak hanya digunakan dalam pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan [25].




**Gambar 2.10** *Unified Modeling Language* Diagram [25]

### 2.11.3 Class Diagram

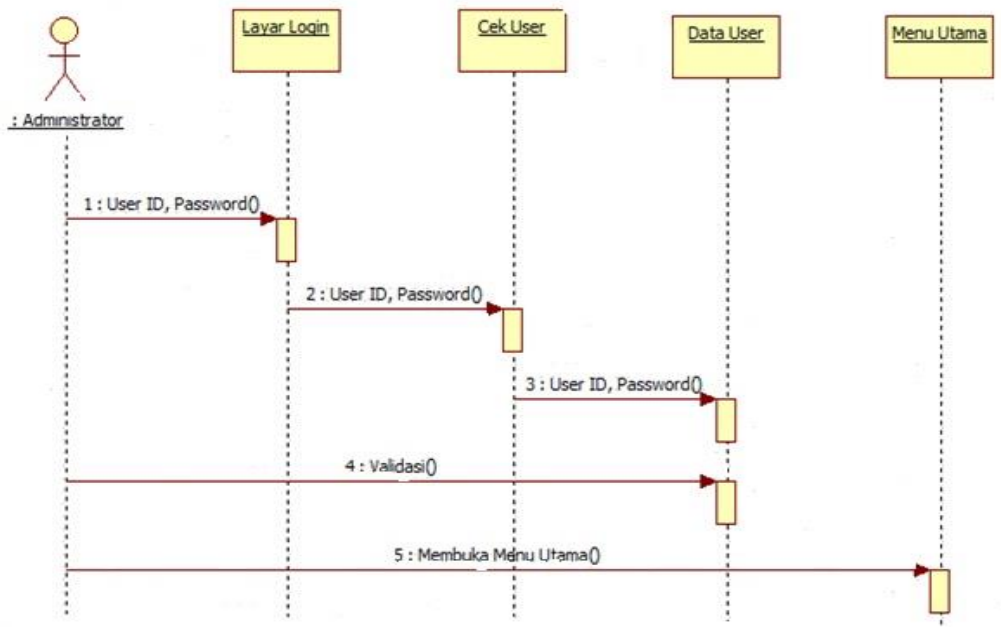
*Class* diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. *Class* diagram memiliki apa yang disebut atribut dan metode atau operasi. *Class* diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan statis yang terdapat di antara mereka. *Class* Diagram juga menunjukkan properti dan operasi sebuah kelas dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut.




**Gambar 2.11 Contoh Class Diagram [25]**

### 2.11.4 Sequence Diagram

*Sequence* diagram merupakan salah satu diagram *interaction* yang menjelaskan bagaimana suatu operasi itu dilakukan, *message* (pesan) apa yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu dan objek-objek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut.




**Gambar 2.12 Contoh Sequence Diagram [25]**

## 2.12 Flowchart

Aliran data yang ada pada suatu sistem dapat digambarkan dengan menggunakan diagram *flowchart*. *Flowchart* atau bagan alir adalah suatu bagan yang berisi simbol – simbol grafis yang menggambarkan arah aliran kegiatan dan data-data yang dimiliki program saat program dijalankan. Pada *flowchart* terdapat tiga komponen yaitu terdapat *input*, proses dan *output*, dan sebagai tambahannya terdapat simbol kondisional yang terletak diantara *input* dan *output*. *Flowchart* pada penelitian ini digunakan untuk menggambarkan perancangan prosedur – prosedur yang terdapat pada sistem *optical character recognition* dan ekstraksi informasi.

## 2.13 Java

Bahasa Java diciptakan oleh James Gosling dan Patrick Naughton dalam suatu proyek dari SUN Microsystem. Pada mulanya ingin diberi nama OAK dari pohon yang terdapat pada kantor James Gosling, tetapi kata OAK telah ada pada sun microsystem, maka diberi nama java (terinspirasi minum kopi). Browser pertama yang dapat membaca java adalah Hot Java. Setelah Browser Netscape dari

perusahaan Netscape Navigator dan IE dari perusahaan Microsoft Inc dapat membaca script java, maka bahasa java semakin popular.

Vendor-vendor lain, seperti IBM, Oracle, Symantec, Inprise (Borland Inc), dan perusahaan-perusahaan mobile, seperti Nokia, Sony Ericsson, Motorola, dan Samsung juga mengadopsi teknologi java.

## 2.14 Neatbeans

NetBeans awalnya dibangun pada tahun 1996 sebagai Xelfi (untuk pemrograman Delphi) oleh seorang mahasiswa dari Charles University di Paraguai. 17 Pada tahun 1997, Roman Stanek membangun sebuah perusahaan dan merilis versi komersial dari NetBeans hingga akhirnya dibeli oleh Sun MicroSystem pada 1999. Hingga saat ini platform NetBeans telah banyak berkembang di bawah SunMicrosystem.

NetBeans merupakan platform framework dan IDE (integrated development environtment) yang digunakan untuk pengembangan aplikasi desktop yang menggunakan Bahasa Java, dan beberapa bahasa lain, seperti Groovy, C, C++ dan banyak lagi. NetBeans IDE dibangun menggunakan Bahasa Java dan dapat dijalankan pada Windows, OS X, Linux, Solaris dan sistem operasi lain yang mendukung JVM. IDE NetBeans merupakan alat pengembangan aplikasi yang terintegrasi.

NetBeans IDE mendukung pengembangan Program yang menggunakan Bahasa Java dari semua versi (Java SE, Java ME, Java EE). Platform Netbeans memperbolehkan pembangunan aplikasi dengan menggunakan modul-modul. Aplikasi yang dibangun menggunakan NetBeans dapat dikembangkan oleh pihak ketiga. Platform NetBeans merupakan platform yang dapat digunakan ulang (reusable) untuk mempermudah pembangunan program menggunakan Bahasa Java.

## 2.15 Perangkat Lunak Pendukung

### 2.15.1 MySQL

Banyak situs web dinamis memerlukan database backend. Database dapat berisi informasi yang ditampilkan dari halaman web kepada pengguna, atau bertujuan dari database mungkin untuk menyimpan informasi yang diberikan oleh pengguna. Dalam beberapa aplikasi, database keduanya menyediakan informasi yang tersedia dan menyimpan informasi baru.

MySQL adalah database yang paling populer untuk digunakan dalam situs web, dikembangkan menjadi cepat dan kecil, khususnya untuk situs web. MySQL sangat populer untuk digunakan dengan situs web yang ditulis dalam bahasa pemrograman PHP, PHP dan MySQL bekerja sama dengan baik [22].

Fitur-fitur MySQL antara lain:

1. ***Relational Database System***. Seperti halnya software database lain yang ada di pasaran, MySQL termasuk RDBMS.
2. **Arsitektur *Client-Server***. MySQL memiliki arsitektur client-server dimana server database MySQL terinstall di server. Client MySQL dapat berada di komputer yang sama dengan server, dan dapat juga di komputer lain yang berkomunikasi dengan server melalui jaringan bahkan internet.
3. **Mengenal perintah SQL standar**. SQL (*Structured Query Language*) merupakan suatu bahasa standar yang berlaku di hampir semua software database. MySQL mendukung SQL versi SQL:2003.
4. **Mendukun Sub Select**. Mulai versi 4.1 MySQL telah mendukung select dalam select (sub select).
5. **Mendukung Views**. MySQL mendukung views sejak versi 5.0.
6. **Mendukung *Stored Prosedured* (SP)**. MySQL mendukung SP sejak versi 5.0.
7. **Mendukung *Triggers***. MySQL mendukung trigger pada versi 5.0 namun masih terbatas. Pengembang MySQL berjanji akan meningkatkan kemampuan trigger pada versi 5.1.
8. Mendukung *replication*.
9. Mendukung transaksi.

10. Mendukung *foreign key*.
11. Tersedia fungsi GIS.
12. Free (bebas didownload).
13. Stabil dan tangguh.
14. Fleksibel dengan berbagai pemrograman.
15. Security yang baik.
16. Dukungan dari banyak komunitas.
17. Perkembangan software yang cukup cepat.

### 2.15.2 XAMPP

XAMPP adalah kit all-in-one yang populer dalam menginstal apache, MySQL, dan PHP dalam satu prosedur. XAMPP juga menginstal phpMyAdmin, aplikasi web yang dapat digunakan untuk mengelola database MySQL [22].

Menurut situs web XAMPP, XAMPP dimaksudkan sebagai lingkungan pengembangan pada komputer lokal. Sebagai lingkungan pengembangan, XAMPP dikonfigurasi untuk menjadi seterbuka mungkin. XAMPP tidak dimaksudkan untuk penggunaan produksi yang tidak aman sebagai lingkungan produksi. Berikut Bagian yang penting dari XAMPP:

1. htdoc, merupakan folder untuk meletakkan file yang akan dijalankan, seperti file Php, HTML, dan script lainnya.
2. phpMyAdmin, merupakan bagian untuk mengelola database MySQL yang ada pada komputer.
3. Control Panel, berfungsi untuk mengelola layanan (service) XAMPP, seperti start service (mulai) atau stop service (berhenti).

XAMPP adalah singkatan dari setiap huruf, yaitu:

1. X: Program ini dapat dijalankan di banyak sistem operasi, seperti Windows, Linux, Mac OS, dan Solaris.
2. A: Apache, server aplikasi web. Tugas utama apache adalah untuk menghasilkan halaman web yang benar kepada pengguna terhadap kode Php yang sudah dituliskan oleh pembuat halaman web.

3. M: MySQL, server aplikasi database. SQL (Structured Query Language) merupakan bahasa terstruktur yang difungsikan untuk mengolah database. MySQL dapat digunakan untuk membuat, mengelola database dan isinya.
4. P: Php, bahasa pemrograman web. Bahasa pemrograman Php (Hypertext Preprocessor) adalah bahasa pemrograman untuk membuat web berbasis server-side scripting. Php digunakan untuk membuat halaman web dinamis.
5. P: Perl, bahasa pemrograman untuk semua tujuan, pertama kali dikembangkan oleh Larry Wall, mesin Unix. Perl dirilis pertama kali tanggal 18 Desember 1987 yang ditandai dengan keluarnya Perl 1. Pada versi-versi selanjutnya, Perl juga tersedia untuk berbagai sistem operasi Unix (SunOS, Linux, BSD, HP-UX), juga tersedia untuk sistem operasi seperti DOS, Windows, PowerPC, BeOS, VMS, EBCDIC, dan PocketPC.

## 2.16 Pengujian Sistem

Pengujian sistem adalah proses pemeriksaan atau evaluasi sistem atau komponen sistem secara manual atau otomatis untuk memverifikasi apakah sistem memenuhi kebutuhan-kebutuhan yang dispesifikasikan atau mengidentifikasi perbedaan-perbedaan antara hasil yang diterapkan dengan hasil yang terjadi [26]. Pengujian seharusnya meliputi tiga konsep berikut.

1. Demonstrasi validitas perangkat lunak pada masing-masing tahap di siklus pengembangan sistem.
2. Penentuan validitas sistem akhir dikaitkan dengan kebutuhan pemakai.
3. Pemeriksa perilaku sistem dengan mengeksekusi sistem pada data sampel pengujian.

Pada dasarnya pengujian diartikan sebagai aktivitas yang dapat atau hanya dilakukan setelah pengkodean (kode program selesai). Namun, pengujian seharusnya dilakukan dalam skala lebih luas. Pengujian dapat dilakukan begitu spesifikasi kebutuhan telah dapat didefinisikan. Evaluasi terhadap spesifikasi dan perancangan juga merupakan Teknik pengujian. Kategori pengujian dapat dikategorikan menjadi dua [26], yaitu:

1. Berdasarkan ketersediaan logic sistem, terdiri dari Black box testing dan White box testing.
2. Berdasarkan arah pengujian, terdiri dari Pengujian top down dan Pengujian bottom up.

## 2.17 Pengujian *Black Box*

Konsep *Black box* digunakan untuk merepresentasikan sistem yang cara kerja didalamnya tidak tersedia untuk diinspeksi. Di dalam black box, item-item yang diuji dianggap "gelap" karena logikanya tidak diketahui, yang diketahui hanya apa yang masuk dan apa yang keluar dari black box [26].

Pada pengujian black box, kasus-kasus pengujian berdasarkan pada spesifikasi sistem. Rencana pengujian dapat dimulai sedini mungkin di proses pengembangan perangkat lunak. Teknik pengujian konvensional yang termasuk pengujian "black box" adalah sebagai berikut [26]:

1. Graph-based testing
2. Equivalence partitioning
3. Comparison testing
4. Orthogonal array testing

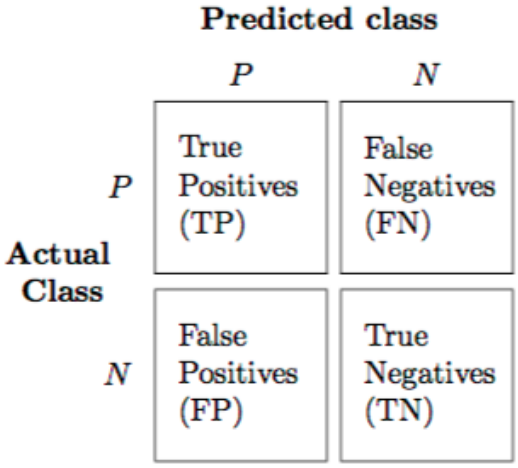
Pada pengujian black box, kita mencoba beragam masukan dan memeriksa keluaran yang dihasilkan. Kita dapat mempelajari apa yang dilakukan kotak, tapi tidak mengetahui sama sekali mengenai cara konversi dilakukan. Teknik pengujian black box juga dapat digunakan untuk pengujian berbasis scenario, dimana isi dalam sistem mungkin tidak tersedia diinspeksi tapi masukan dan keluaran yang didefinisikan dengan use case dan informasi analisis yang lain.

## 2.18 Pengujian Akurasi

Terdapat beberapa metode pengujian, salah satu metode yang dapat digunakan untuk mengukur kinerja suatu metode klasifikasi adalah dengan membandingkan nilai true value. Pada dasarnya metode pengujian ini mengandung informasi yang membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan klasifikasi yang seharusnya.

### 2.18.1 *Confusion Matrix*

*Confusion Matrix* merupakan sebuah tabel yang menggambar jumlah data uji yang terklasifikasi dengan benar dan jumlah data uji yang terklasifikasi dengan salah. *Confusion Matrix* juga adalah matriks yang didalamnya terdapat jumlah dari *True Positive, True Negative, False Positive,* dan *False Negative* [27]. TP dan TN merupakan hasil klasifikasi yang benar sedangkan FP dan FN merupakan hasil klasifikasi yang salah. Adapun gambaran *confusion matrix* dapat dilihat pada gambar berikut:




**Gambar 2.13 *Confusion Matrix***

a. Akurasi merupakan persentase dari total hasil prediksi yang bernilai true dari semua data. Adapun untuk melakukan perhitungan tingkat akurasi menggunakan persamaan berikut.

$$Akurasi = \frac{TP}{TP + FN + FP + TN} \text{ x } 100\% \qquad \textbf{(2.53)}$$

Keterangan:

TP : *True Positive*

TN : *True Negative*

FP : *False Positive*

FN : *False Negative*

b. *Precision* adalah nilai ketepatan dari hasil suatu model. Persamaannya menggunakan perbandingan antara hasil *true positive* dengan total data dengan label *positive*. Adapun untuk perhitungan *precision* menggunakan persamaan berikut[28].

$$Precision = \frac{TP}{TP + FP} \quad \textbf{(2.54)}$$

c. *Recall* adalah nilai kelengkapan dari sebuah model. Persamaan *recall* menggunakan perbandingan antara *true positive* terhadap total contoh yang benar-benar *positive*. Adapun untuk perhitungan *recall* menggunakan persamaan berikut[28].

$$Recall = \frac{TP}{TP + FN} \quad \textbf{(2.55)}$$

d. *F-measure* merupakan perhitungan untuk mencari nilai *harmonic mean* dari *precision* dan *recall,* Adapun untuk perhitungan *recall* menggunakan persamaan berikut[28].

$$F1\ Score = \frac{2 * precision * recall}{precision + recall} \quad \textbf{(2.56)}$$

Jika dalam suatu pengujian jumlah kelas yang diklasifikasikan memiliki lebih dari dua kelas, maka rumus untuk menghitung *presicion*, dan *recall* adalah sebagai berikut [29] :

$$Precision = \frac{\sum_i^L \frac{TP_i}{TP_i + FP_i}}{L} X\ 100\% \quad \textbf{(2.57)}$$

$$Recall = \frac{\sum_i^L \frac{TP_i}{TP_i + FN_i}}{L} X\ 100\% \quad \textbf{(2.58)}$$

Keterangan

L : Jumlah Keseluruhan Data Yang Diuji/Jumlah Kelas

### 2.18.2 Classification Accuracy

*Classification Accuracy* merupakan sebuah metode yang digunakan untuk menghitung jumlah kebenaran suatu *classifier* dengan menampilkan persentase dari data yang berhasil di klasifikasikan[30]. Adapun untuk perhitungannya menggunakan persamaan berikut:

$$Accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ predictions\ made}\ x\ 100\% \quad \textbf{(2.59)}$$

Keterangan

*Number of correct Prediction* : Jumlah data yang berhasil di kelompokkan dari *classifier*

*Total number of predictions made* : Total data yang digunakan pada proses klasifikasi

Jika *classifier* yang diuji memiliki jumlah lebih dari dua, maka untuk mencari rata-rata dari setiap *classifier* menggunakan persamaan sebagai berikut:

$$Classifier\ = \frac{\sum_{i=1}^{n} x_i}{n}\ x\ 100\% \quad \textbf{(2.60)}$$

Keterangan

$x_i$ : Nilai akurasi dari *classifier* ke - i

n : Jumlah total yang data uji