BAB 2
LANDASAN TEORI

2.1 Berita

Untuk dapat mengenal informasi, dibutuhkan upaya menuntun ke mana dan bagaimana memperoleh fakta yang diperlukan. Berita adalah segala laporan mengenai peristiwa, kejadian, gagasan, fakta yang menarik perhatian dan penting untuk disampaikan atau dimuat dalam media massa agar diketahui atau menjadi kesadaran umum [4].

Dengan demikian, jika diamati dari semua definisi tersebut pada dasarnya berita mengandung beberapa unsur antara lain :

1. Suatu peristiwa, kejadian, gagasan, pikiran, fakta yang aktual.
2. Menarik perhatian karena ada factor yang luar biasa (*extraordinary*) di dalamnya.
3. Penting.
4. Dilaporkan, diumumkan, atau dibuat untuk menjadi kesadaran umum supaya menjadi pengetahuan bagi orang banyak (massa).
5. Laporan itu dimuat di media tertentu.

Dari kelima unsur di atas dapat disimpulkan bahwa suatu peristiwa, kejadian, gagasan, atau yang disebut dengan "fakta" betapa pun aktual, menarik, dan pentingnya, jika tidak dilaporkan atau diberitakan melalui media massa dan tidak disampaikan kepada umum untuk diketahui, hal tersebut bukanlah berita. Artinya, fakta menjadi berita bila dilaporkan [4].

Berita yang didasarkan pada topik masalah mencakup berbagai bidang yang sangat kompleks. Secara besarannya biasa dikelompokan menjadi berita politik, ekonomi, sosial budaya, hukum , olahraga, militer, kriminal, atau kejahatan dan sebagainya. Bidang – bidang tersebut masih bisa dirinci lagi [4].

Demikian rumitnya, demi menggali unsur daya tariknya, pengelompokan bidang di atas dibuat semacam rubrikasi dalam surat kabar dan majalah. Mulai dari persoalan wanita, gaya hidup, menu makanan, agama, hingga rubrik anak dan remaja. Semua ini bergantung pada visi dan misi perusahaan media terkait [4].

Dalam dunia maya, sebagai media baru dalam komunikasi massa, untuk situs web beritadibuat dengan nama manajemen konten yang berlapis – lapis jendela atau navigasi untuk masuk kepada menunya (*feeding*) atau bahkan lokasi beritanya.

Berikut dibahas beberapa bidang yang menyangkut berita dilihat dari segi persoalan sebagai gambaran tentang luasnya bidang – bidang pemberitaan dan sumber – sumber yang harus diliput oleh seorang wartawan.

1. Politik

   Berita politik adalah berita yang menyangkut kegiatan politik atau peristiwa disekitar masalah – masalah ketatanegaraan dan segala hal yang berhubungan dengan urusan pemerintah dan negara. Politik di sini harus diartikan sebagai upaya manusia untuk menata kehidupan rakyat, pemerintahan, dan Negara demi mencapai suatu tujuan dan cita – cita bersama yang luhur, yaitu perbaikan hidup atau nasib bangsanya.

   Berita politik menjadi menu pokok isi media karena pengaruhnya cukup luas dan mendalam bagi kehidupan rakyat sehari-hari. Berita politik akan terasa meninggi intensitas dan gemanya tatkala menjelang, saat, dan pasca – pemilihan umum, siding – siding DPR/MPR, pembentukan cabinet, krisis lembaga legislative, pemilihan kepala daerah, kongres partai, dan sebagainya.

2. Ekonomi

   Selain berita politik, masalah ekonomi juga memiliki pengaruh besar terhadap kehidupan kita. Masalah ekonomi sering kali memberi dampak nyata bagi kehidupan manusia. Berita ekonomi sebenarnya mencakup aspek yang sangat luas, yaitu perdagangan, finansial, perindustrian, perdagangan, pertambangan, perbankan, tenaga kerja, dunia usaha, valuta asing, dan pasar modal.

3. Teknologi

   Pada dasarnya manusia menyukai kemajuan. Oleh karena itu, berita tentang perkembangan atau kemajuan teknologi juga menarik perhatian pembaca. Keseharian kita tidak lepas dari produk teknologi. Mulai dari

telepon genggam, alat transportasi, robot, hingga gaya hidup. Semuanya merupakan produk dari ilmu pengetahuan dan teknologi.

4. Kesehatan

   Kesehatan merupakan keadaan dimana setiap manusia mampu untuk hidup produktif tanpa adanya gangguan dalam kesehatan. Bertia kesehatan meliputi seluruh kegiatan dalam pemeliharaan kesehatan. Pencegahan, pengobatan, perawatan, kebiasaan yang merusak kesehatan, dan sebagainya.

5. Pariwisata

   Berita periwisata meliputi seluruh kegiatan yang berhubungan dengan perjalanan wisata, daya tarik wisata, pengusaha objek wisata, keindahan tempat wisata, dan sebagainya.

6. Olahraga

   Berita olahraga meliputi seluruh kegiatan olahraga termasuk cabang – cabang olahraga seperti atletik, renang, senam, balap sepeda, balap mobil/motor, tinju, gulat, renang, yudo, karate, silat, tennis, bulu tangkis, bola, polo air, golf, tenis meja, dan sebagainya, baik yang berskala local, regional, nasional, maupun internasional.

2.2 Klasifikasi

Klasifikasi merupakan suatu pekerjaan menilai objek data untuk memasukannya ke dalam kelas tertentu dari sejumlah kelas yang tersedia. Dalam klasifikasi ada dua pekerjaan utama yang dilakukan. Pertama, pembangunan model sebagai prototipe untuk disimpan sebagai memori. Kedua, penggunaan model untuk melakukan pengenalan/klasifikasi/prediksi pada suatu objek lain, agar diketahui di kelas mana objek data tersebut dalam model yang sudah disimpannya [1].

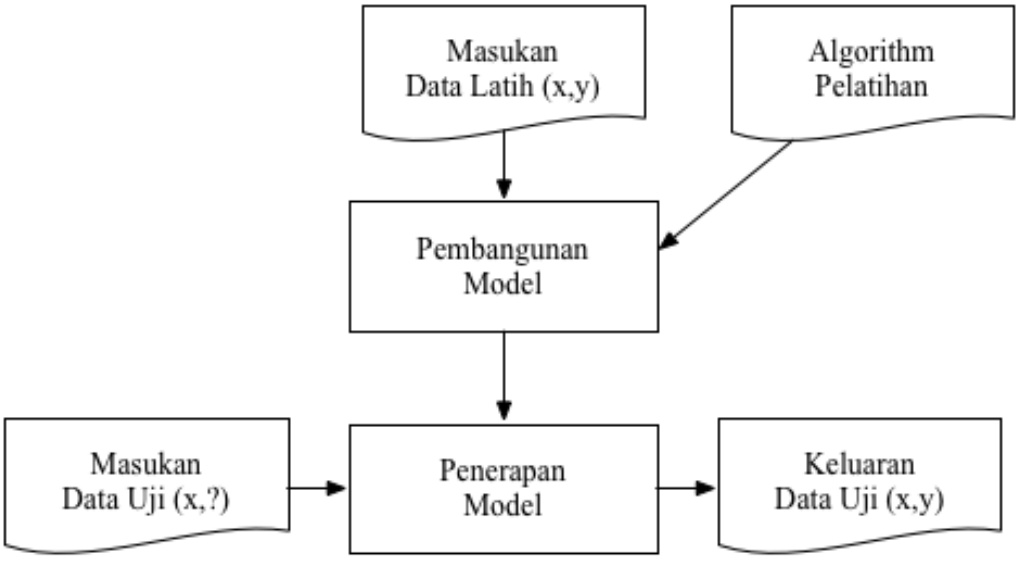
Contoh aplikasi yang sering ditemui adalah pengklasifikasian jenis hewan, yang mempunyai sejumlah atribut. Dengan atribut tersebut, jika ada hewan baru, kelas hewannya dapat langsung diketahui. Contoh lain adalah bagaimana melakukan diagnosis penyakit kulit kanker melanoma (Amaliyah *et al*, 2011),

yaitu dengan melakukan pembangunan model berdasarkan data latih yang ada, kemudian menggunakan model tersebut untuk mengidentifikasi penyakit pasien baru sehingga diketahui apakah pasien tersebut menderita kanker atau tidak [1].

2.2.1 Model

Klasifikasi dapat didefinisikan sebagai perkerjaan yang melakukan pelatihan/pembelajaran terhadap fungsi target f yang memetakan setiap set atribut (fitur) x ke satu dari sejumlah label kelas y yang tersedia. Pekerjaan pelatihan tersebut akan menghasilkan suatu model yang kemudian disimpan sebagai memori [1].

Model dalam klasifikasi mempunyai arti yang sama dengan kotak hitam, di mana ada suatu model yang menerima masukan, kemudian mampu melakukan pemikiran terhadap masukan tersebut, dan memberikan jawaban sebagai keluaran dari hasil pemikirannya. Kerangka kerja klasifikasi ditunjukan pada Gambar 2.1. Pada gambar tersebut disediakan sejumlah data latih (x,y) untuk digunakan sebagai data pembangunan model. Model tersebut kemudian diapakai untuk memprediksi kelas dari data uji (x,?) sehingga diketahui kelas y yang sesungguhnya.



Gambar 2.1 Proses Pekerjaan Klasifikasi

Kerangka kerja seperti yang ditunjukan pada Gambar 2.1 meliputi dua langkah proses, yaitu induksi dan deduksi. Induksi merupakan langkah untuk

membangun model klasifikasi dari data latih yang diberikan, disebut juga proses pelatihan, sedangkan deduksi merupakan langkah untuk menerapkan model tersebut pada data uji sehingga kelas yang sesungguhnya dari data uji dapat diketahui [1].

2.2.2 Pengukuran Kinerja Klasifikasi

Sebuah sistem yang melakukan klasifikasi diharapkan dapat melakukan klasifikasi semua set data dengan benar, tetapi tidak dapat dipungkiri bahwa kinerja suatu sistem tidak dapat 100% benar sehingga sebuah sistem klasifikasi juga harus diukur kinerjanya. Umumnya, pengukuran kinerja klasifikasi dilakukan dengan matriks konfusi[1].

Matriks konfusi merupakan tabel pencatat hasil kerja klasifikasi. Tabel 2.1 merupakan contoh matriks konfusi yang melakukan klasifikasi masalah biner (dua kelas), hanya ada dua kelas, yaitu kelas 0 dan kelas 1. Setiap sel f_{ij} dalam matriks menyatakan jumalah data dari kelas I yang hasil prediksinya masuk ke kelas j. Misalnya, sel f_{11} adalah jumlah data dalam kelas 1 yang secara benar dipetakan ke kelas 1, dan f_{10} adalah data dalam kelas 1 yang dipetakan secara sah ke kelas 0.

Tabel 2.1 Matriks Konfusi untuk klasifikasi dua kelas

f_{ij}		Kelas hasil prediksi (j)	
		Kelas = 1	Kelas = 0
Kelas asli (i)	Kelas = 1	f_{11}	f_{10}
	Kelas = 0	f_{01}	f_{00}

Dengan mengetahui jumlah data yang diklasifikasikan secara benar, kita dapat mengetahui akurasi hasil prediksi, dan dengan mengetahui jumlah data yang diklasifikasikan secara sah, kita dapat mengetahui laju eror dari prediksi yang dilakukan. Dua kuantitas ini digunakan sebagai metric kinerja klasifikasi.

Untuk menghitung akurasi digunakan formula

$$\text{Akurasi} = \frac{\text{Jumlah data yang diprediksi secara benar}}{\text{jumlah prediksi yang dilakukan}} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

Untuk menghitung lahu error (kesalahan prediksi) digunakan formula

$$\text{Laju eror} = \frac{\text{Jumlah data yang diprediksi secara salah}}{\text{jumlah prediksi yang dilakukan}} = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

Semua algoritma klasifikasi berusaha membentuk model yang mempunyai akurasi tinggi atau laju eror yang rendah. Umumnya, model yang dibangun dapat memprediksi dengan benar pada semua data yang menjadi data latihnya, tetapi ketika model berhadapan dengan data uji, barulah kinerja model dari sebuah algoritma klasifikasi ditentukan [1].

2.3 Praproses Dokumen

Dokumen Preparation atau bisa disebut juga dengan dokumen preprocessing adalah suatu proses/langkah yang dilakukan untuk membuat data mentah menjadi data yang berkualitas. Proses persiapan dokumen meliputi *case folding*, tokenisasi, dan *stopwords*. Keempat proses ini perlu dilakukan sebelum proses *feature extraction*.

2.3.1 Case Folding

Case folding adalah mengubah semua huruf dalam dokumen menjadi huruf kecil. Hanya huruf "a" sampai dengan "z" yang diterima. Karakter selain huruf dihilangkan dan dianggap delimiter.

2.3.2 Tokenisasi

Secara garis besar tokenisasi adalah tahap memecah sekumpulan karakter dalam suatu teks kedalam satuan kata. Sekumpulan karakter tersebut dapat berupa karakter whitespace, seperti enter, tabulasi, spasi.

2.3.3 Stopwords

Stopword adalah katakata yang tidak deskriptif yang dapat dibuang dalam pendekatan bag-of-words. Contoh stopwords adalah "yang", "dan", "di", "dari" dan seterusnya.

2.4 Text Feature Extraction

Text Feature Extraction adalah sebuah proses untuk mengambil atau mengekstrak karakteristik atau informasi yang penting dari suata data untukdigunakan dalam proses klasifikasi.

2.4.1 TF-IDF

Metode TF-IDF merupakan metode untuk menghitung bobot setiap kata yang paling umum digunakan pada information retrieval. Metode ini juga terkenal efisien, mudah dan memiliki hasil yang akurat [6]. Metode ini akan menghitung nilai Term Frequency (TF) dan Inverse Document Frequency (IDF) pada setiap token (kata) di setiap dokumen dalam korpus. Metode ini akan menghitung bobot setiap token t di dokumen d dengan rumus:

$$\text{idf}(t) = \text{Log}\left(\frac{D}{df}\right) \quad (2.1)$$

$$tf\text{-}idf(t) = tf(t,d) * idf(t) \quad (2.2)$$

Dimana :

d : dokumen ke-d

t : kata ke-t dari kata kunci

tf-idf (t): bobot dokumen ke-d terhadap kata ke-t

tf : banyaknya kata yang dicari pada sebuah dokumen

IDF : Inversed Document Frequency

D : total dokumen

df : banyak dokumen yang mengandung kata yang dicari

Berdasarkan rumus diatas, berapapun besarnya nilai *tf* (t,d), apabila D = df maka akan didapatkan hasil 0 (nol) untuk perhitungan idf. Untuk itu dapat ditambahkan nilai 1 pada sisi idf, sehingga perhitungan bobotnya menjadi sebagai berikut:

$$\text{idf}(t) = \text{Log}\left(\frac{D}{df}\right) + 1 \quad (2.1)$$

Rumus (2.1) dapat dinormalisasi dengan Rumus (2.2) dengan tujuan untuk menstandarisasi nilai bobot ke dalam interval 0 s.d. 1, sbb:

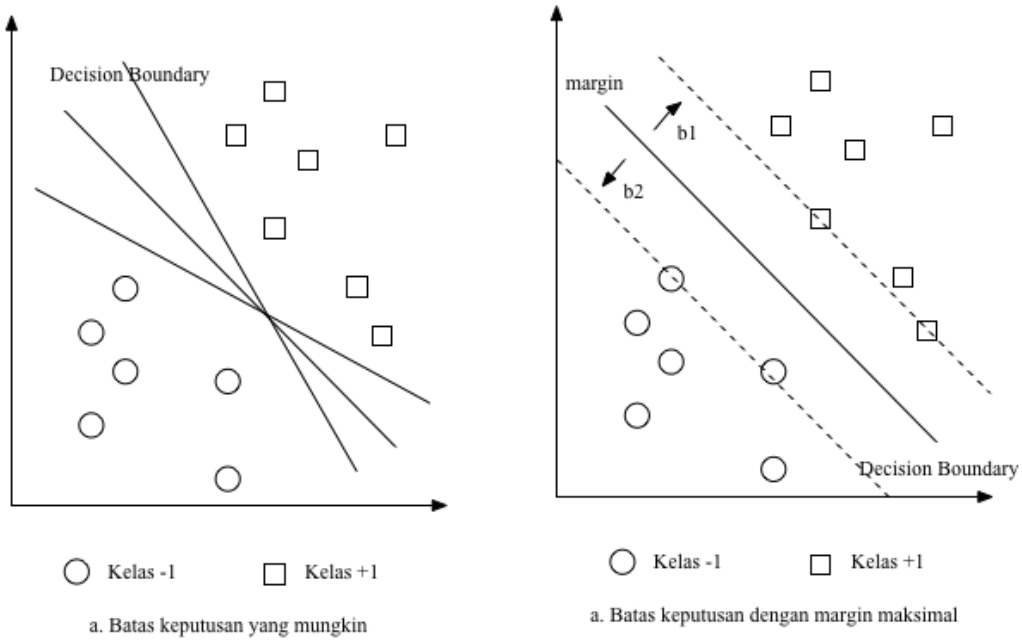
Rumus Tf-Idf dengan menggunakan normalisasi

$$w_{ij} = \frac{tf*(\log\left(\frac{D}{df}\right)+1)}{\sqrt{\sum_{k=1}^{t}(tf_{ik})^2 * [(\log\left(\frac{D}{df}\right)+1)]^2}} \quad (2.3)$$

2.5 Support Vector Machine (SVM)

2.5.1 Konsep SVM

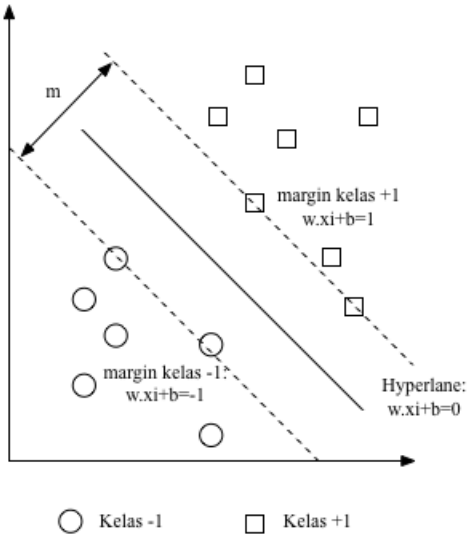
Ide dasar SVM adalah memaksimalkan batas *hyperlane*, yang diilustrasikan pada Gambar 2.2. Pada gambar (a) ada sejumlah pilihan *hyperlane* yang mungkin untuk set data, sedangkan gambar (b) merupakan *hyperlane* dengan margin maksimal. Meskipun sebenarnya pada gambar (a) bisa juga menggunakan *hyperlane* sembarang, tetapi *hyperlane* dengan margin yang maksimal akan memberikan generalisasi yang lebih baik pada metode klasifikasi[1].



Gambar 2.2 Batas keputusan yang mungkin untuk set data

Konsep klasifikasi dengan SVM dapat dijelaskan secara sederhana sebagai usaha mencari *hyperlane* terbaik yang berfungsi sebagai pemisah dua buah kelas data pada input space (Nugroho, 2007). Gambar 2.2 memperlihatkan beberapa data yang merupakan anggota dari duah buah kelas data, yaitu +1 dan -1. Data yang tergabung pada kelas -1 disimbolkan dengan bentuk lingkaran, sedangkan data pada kelas +1, disimbolkan dengan bujur sangkar[1].

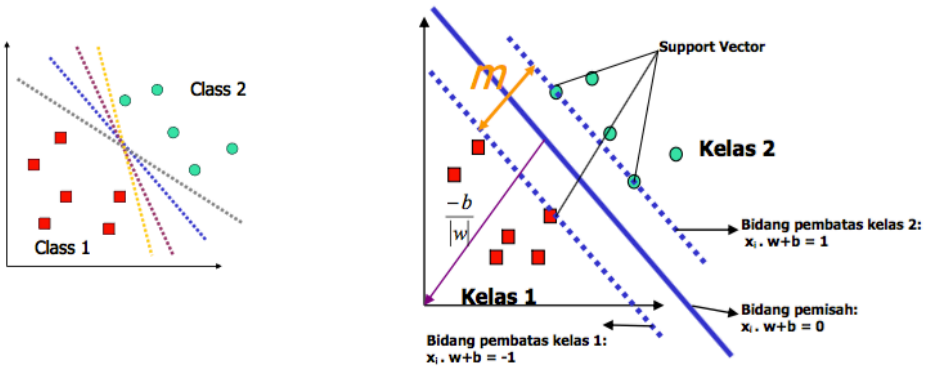
Hyperlane (batas keputusan) pemisah terbaik antara kedua kelas dapat ditemukan dengan mengukur margin *hyperlane* tersebut dan mencari titik maksimalnya. Margin adalah jarak antara *hyperlane* tersebut dengan data terdekat dari masing – masing kelas. Data yang paling dekat ini disebut sebagai *support vector*. Garis solid pada gambar 2.2 (b) sebelah kanan menunjukan *hyperlane* yang terbaik, yaitu yang terletak tepat ditengah – tengah kedua kelas, sedangkan data lingkaran dan bujur sangkar yang dilewati garis batas margin (garis putus-putus) adalah *support vector*. Usaha untuk mencari lokasi *hyperlane* ini merupakan inti dari proses pelatihan pada svm[1].



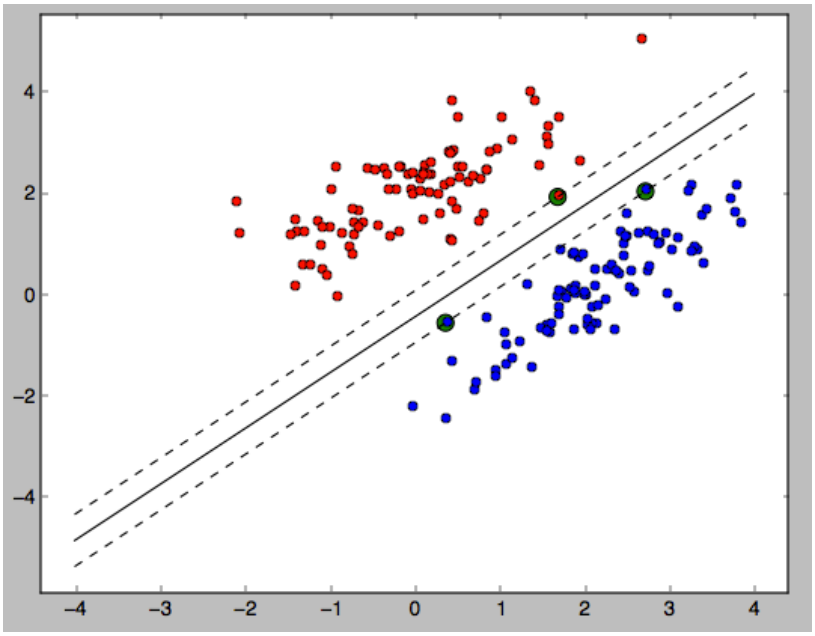
Gambar 2.3 Margin *Hyperplane*

2.5.2 SVM pada *Linearly Separable Data*

Linearly separable data merupakan data yang dapat dipisahkan secara linier. Misalkan $\{x_1 ,..., x_n\}$adalah dataset dan $y_i \in \{+1,-1\}$ adalah label kelas dari data x_i. Pada Gambar 2.4 dapat dilihat berbagai alternatif bidang pemisah yang dapat memisahkan semua data set sesuai dengan kelasnya. Namun, bidang pemisah terbaik tidak hanya dapat memisahkan data tetapi juga memiliki margin paling besar[5].



Gambar 2.4 Alternatif bidang pemisah (kiri) dan bidang pemisah terbaik dengan margin (m) terbesar (kanan).



Gambar 2.5 diagram plot data linear

Setiap data latih dinyatakan oleh (*xi,yi*), di mana *i* = 1,2, ..., N, dan *xi* = $\{xi1, xi2, \ldots, xiq\}^T$ merupakan atribut (fitur) set untuk data latih ke –*i*. yi ∈ {-1,+1} menyatakan label kelas. Kelas 1 merupakan kelas -1 dan kelas 2 merupakan kelas +1. *Hyperplane* klasifikasi linear SVM, seperti pada Gambar 2.4, dinotasikan dengan

$$\mathbf{w}.\boldsymbol{x_i} + b = 0 \qquad (2.4)$$

w adalah normal bidang dan b adalah posisi bidang relatif terhadap pusat

koordinat. $\mathbf{W} \cdot \boldsymbol{X_i}$ merupakan inner product dalam antara *w* dan x_i. Data x_i yang masuk ke dalam kelas -1 adalah data yang memenuhi pertidaksamaan berikut:

$$\mathbf{w}.\boldsymbol{x_i} + b \leq -1 \quad (2.5)$$

Sementara data x_i yang masuk ke dalam kelas +1 adalah data yang memenuhi pertidaksamaan berikut:

$$\mathbf{w}.\boldsymbol{x_i} + b \geq +1 \quad (2.6)$$

Sesuai dengan Gambar 2.4, jika ada dalam kelas -1 (misalnya, x_a) yang bertempat di *hyperplane*, Persamaan 2.2 akan terpenuhi. Untuk data kelas -1 dinotasikan dengan

$$\mathbf{w}.\boldsymbol{x_a} + b = 0 \quad (2.7)$$

Sementara kelas +1 (misalnya, x_b) akan memenuhi persamaan

$$\mathbf{w}.\boldsymbol{x_b} + b = 0 \quad (2.8)$$

x_b - x_a adalah vektor paralel di posisi *hyperplane* dan diarahkan dari x_a ke x_b. Karena inner produk dalam bernilai 0, arah w harus tegak lurus terhadap *hyperplane* (bidang pemisah Gambar 2.4).

Dengan memberikan label -1 untuk kelas 1 dan +1 untuk kelas 2, prediksi semua data uji akan menggunakan formula

$$y = +1 \text{ , jika } w.x_i + b > 0$$

$$y = -1 \text{ , jika } w.x_i + b < 0 \quad (2.9)$$

Sesuai dengan Gambar 2.4, *hyperplane* untuk kelas -1 (garis putus – putus) adalah data pada *support vector* yang memenuhi persamaan

$$w.x_a + b = -1 \quad (2.10)$$

Sementara hyperplane untuk kerlas +1 (garis putus – putus) memenuhi persamaan

$$w.x_b + b = 1 \quad (2.11)$$

Dengan demikian margin dapat dihitung dengan mengurangi Persamaan 2.8 dengan 2.7, didapatkan

$$w.(x_b - x_a) = 2 \quad (2.12)$$

Margin *hyperplane* diberikan oleh jarak antara dua bidang pembatas.

Notasi di atas diringkas menjadi

$$\|w\| \text{ x d atau d} = \frac{2}{\|w\|} \tag{2.13}$$

2.5.3 *Hyperplane* SVM

Klasifikasi kelas data pada SVM pada Persamaan 2.5 dan 2.6 dapat digabungkan dengan notasi

$$y_i(\text{w}.x_i + \text{b}) \geq 1,\ i = 1,2,\ldots,\text{N} \tag{2.14}$$

Margin optimal dihitung dengan memaksimalkan jarak antara hyperplane dan data terdekat. Jarak ini dirumuskan dengan Persamaan 2.10 ($\|w\|$ adalah vektor bobot w). Selanjutnya, masalah ini diformulasikan ke dalam *quadratic programming (QP) problem*, dengan meminimalkan invers Persamaan 2.10, $\frac{1}{2}\|w\|^2$, dibawah kendala(syarat), seperti berikut:

$$\text{Minimalkan : } \frac{1}{2}\|w\|^2 \tag{2.15}$$

dengan syarat seperti pada notasi Persamaan 2.11

Optimalisasi ini dapat diselesaikan dengan Lagrange Multiplier:

$$Lp = \frac{1}{2}\| w \|^2 - \sum_{i=1}^{N} a_i y_i (\text{w} . \text{x}_i + \text{b}) - 1 \tag{2.16}$$

a_i adalah Lagrange Multiplier yang berkorespondensi dengan x_i. Nilai a_i adalah nol atau positif. Untuk meminimalkan Lagrangian, Persamaan 2.13 harus diturunkan pada *w* dan *b*, dan diset dengan nilai 0 untuk syarat optimalisasi di atas.

Syarat 1:

$$\frac{\partial Lp}{\partial w} = 0,\ \text{w} = \sum_{i=1}^{N} a_i\ y_i\ x_i \tag{2.17}$$

Syarat 2:

$$\frac{\partial Lp}{\partial w} = 0,\ \sum_{i=1}^{N} a_i\ y_i = 0 \tag{2.18}$$

N adalah jumlah data yang menjadi *support vector*. Karena Lagrange Multiplier (α) tidak diketahui nilainya, persamaan di atas tidak dapat diselesaikan secara langsung untuk mendapatkan *w* dan *b*. Untuk menyelesaikan masalah

tersebut, modifikasilah Persamaan 2.13 menjadi kasus pemaksimalan, syarat optimal untuk dualitasnya menggunakan kendala (*constraint*) Karush-Kuhn-Tucker (KKT) berikut:

Syarat 1:

$$a_i[y_i\ (w.x_i + b) - 1] = 0 \tag{2.19}$$

Syarat 2:

$$a_i \geq 0,\ \text{i} = 1,2,\ldots,\text{N} \tag{2.20}$$

Dengan menerapkan kendala pada Persamaan 2.16 dan 2.17, dipastikan bahwa nilai Lagrange Multiplier sama banyaknya dengan data latih, tetapi sebenarnya banyak dari data latih yang Lagrange Multiplier-nya sama dengan nol (karena hanya beberpa saja yang akan menjadi support vector) ketika menerapkan syarat pertama. Kendala diatas menyatakan bahwa Lagrange Multiplier a_i harus nol, kecuali untuk data latih x_i yang memenuhi persamaan

$$y_i\ (w.x_i + b) = 1 \tag{2.21}$$

Data latih itu, dengan a_i > 0, terletak pada *hyperplane* b_{i1} atau b_{i2}, dan disebut *support vector*. Data latih yang tidak terletak di *hyperplane* tersebut mempunyai a_i = 0. Persamaan 2.14 dan 2.15 juga menyarankan parameter *w* dan *b*, yang mendefinisikan hyperplane, hanya tergantung *support vector*.

Walaupun bisa diselesaikan, masalah optimalisasi di atas masih sulit karena banyaknya parameter: w, b, dan a_i. Untuk menyederhanakannya, persamaan optimalisasi 2.13 harus ditransformasi ke dalam fungsi Lagrange Multiplier itu sendiri (disebut dualitas masalah).

Persamaan Lagrange Multiplier 2.13 dapat dijabarkan menjadi

$$Lp = \frac{1}{2}\| w \|^2 - \sum_{\text{i}=1}^{\text{N}} \text{a}_\text{i}\text{y}_\text{i}(\text{w} \,.\, \text{x}_\text{i}) - b\sum_{\text{i}=1}^{\text{N}} \text{a}_\text{i}\text{y}_\text{i} + \sum_{\text{i}=1}^{\text{N}} \text{a}_\text{i} \tag{2.22}$$

Syarat optimal 2.15 ada dalam suku ketiga di ruas kanan dalam Persamaan 2.18, dan memaksa suku ini menjadi sama dengan 0. Dengan mengganti *w* dari syarat 2.14, dan suku $||w||^2 = w_i\ .\ w_j$, persamaan di atas akan berubah menjadi dualitas Lagrange Multiplier yang berupa Ld, didapatkan:

Maksimalkan:

$$Ld = \sum_{i=1}^{N} a_i - \frac{1}{2} \sum_{i,j} a_i a_j y_i y_j x_i . x_j \quad (2.23)$$

Syarat 1:

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

Syarat 2:

$$a_i \geq 0, \text{ i} = 1,2, ..., \text{N} \quad (2.25)$$

$\boldsymbol{x_i} \boldsymbol{.} \boldsymbol{x_j}$ merupakan dot – product dua data dalam data latih. Hyperplane (batas keputusan atau pemisah) didapatkan dengan formula

$$\left(\sum_{i=1}^{N} a_i y_i x_i . z \right) + b = 0 \quad (2.26)$$

N adalah jumlah data yang menjadi support vector, x_i merupakan support vector, z merupakan data uji yang akan diprediksi kelasnya, dan x_i .z merupakan inner-product antara x_i dan z. Untuk nilai b didapatkan dari Persamaan 2.16 pada support vector. Karena a_i dihitung dengan metode numerik dan mempunyai eror numerik, nilai yang dihitung untuk *b* bisa jadi tidak sama. Hal ini disebabkan oleh support vector yang digunakan dalam Persamaan 2.16, biasanya diambil nilai rata – rata dari *b* yang didapat untuk menjadi parameter *hyperplane*. Untuk mendapatkan b. Persamaan 2.16 dapat disederhanakan menjadi

$$b_i = 1 - y_i(w. x_i) \quad (2.27)$$

Penjelasan di atas berdasarkan asumsi bahwa kedua kelas dapat terpisah secara sempurna oleh *hyperplane*. Akan tetapi, pada umumnya kedua kelas tersebut tidak dapat terpisah secara sempurna. Hal ini menyebabkan proses optimalisasi tidak dapat diselesaikan karena tidak ada w dan b yang memenuhi Pertidaksamaan 2.11. Untuk itu pertidaksamaan itu dimodifikasi dengan memasukan variabel slack ξ ($\xi_i \geq 0$). Penambahan variabel ξ sering juga disebut *soft margin hyperplane*.Hasilnya adalah

$$y_i (w. x_i + b) \geq 1 - \xi_i \quad (2.28)$$

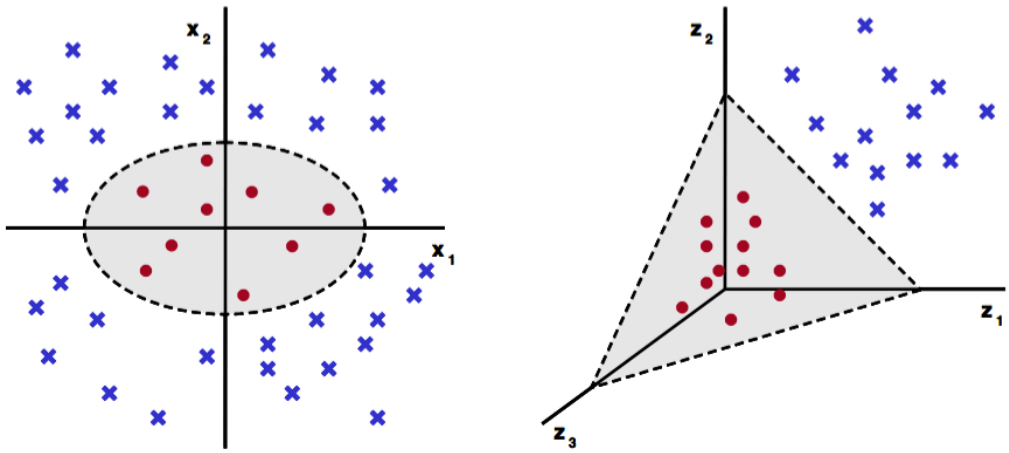
Demikian juga untuk masalah Persamaan 2.12:

$$\frac{1}{2}\| w \|^2 + C\sum_{i=1}^{N} \xi_i \tag{2.29}$$

Metode untuk mengoptimalisasi *hyperplane* SVM umumnya dipakai untuk menyelesaikan pemrograman kuadratik dengan kendala yang ditetapkan. Beberapa pilihan metode yang bisa digunakan adalah *chunking* (Vapnik, 1982), metode dekomposisi (Osuna, 1997), dan *sequential minimal optimization* (SMO)(Plat, 1999).

2.5.4 SVM pada *NonLinearly Separable Data*

SVM sebenarnya adalah hyperplane linear yang hanya bekerja pada data yang dapat dipisahkan secara linear. Untuk data yang distribusi kelasnya tidak linear biasanya digunakan pendekatan kernel pada fitur data dari awal set data. Kernel dapat di definisikan sebagai suatu fungsi yang memetakan fitur data dari dimensi awal (rendah) ke fitur lain yang berdimensi lain yang lebih tinggi (bahkan jauh lebih tinggi). Pendekatan ini berbeda dengan metode klasifikasi pada umumnya yang justru mengurangi dimensi awal untuk menyederhanakan proses komputasi dan memberikan akurasi prediksi yang lebih baik.



a. Data dalan fitur dimensi rendah b. Data dalam fitur dimensi tinggi

Gambar 2.6 Dimensi Data

Ilustrasi kernel yang digunakan untuk memetakan dimensi awal (yang lebih rendah) set data ke dimensi baru (yang relatif lebih tinggi) ada di Gambar

2.5. Algoritma pemetaan kernel ditunjukan pada algoritma 2.1.

Algoritma 2.1 Pemetaan dengan kernel

$\emptyset : D^q \longrightarrow D^r$

x $\longrightarrow$ $\emptyset$(x)

$\emptyset$ merupakan fungsi kernel yang digunakan untuk pemetaan, D merupakan data latih, q merupakan set fitur dalam satu data yang lama, dan r merupakan set fitur yang baru sebagai hasil pemetaan untuk setiap data latih. Sementara x merupakan data latih, di mana x1, x2, ..., $x_n \in D^q$ merupakan fitur – fitur yang akan dipetakan ke fitur berdimensi tinggi r, jadi untuk set data yang digunakan sebagai pelatihan dengan algoritma yang ada dari dimensi fitur yang lama D ke dimensi baru r. Misalnya, untuk n sample data:

$$(\emptyset(x_1), y_1, \emptyset(x_2), y_2, ..., \emptyset(x_n), y_n) \in D^r \quad (2.30)$$

Selanjutnya dilakukan proses pelatihan yang sama sebagaimana pada SVM linear. Proses pemetaan pada fase ini memerlukan perhitungan dot – product dua buah data pada ruang fitur baru. Dot – product kedua buah vektor (x_i) dan (x_j) dinotasikan sebagai $\emptyset(x_i)$. $\emptyset(x_j)$. Nilai dot – product kedua buah vektor ini dapat dihitung secara tidak langsung, yaitu tanpa mengetahui fungsi transformasi $\emptyset$. Teknik komputasi seperti ini kemudian disebut trik kernel, yaitu menghitung dot – product dua buah vektor di ruang dimensi baru dengan memakai komponen kedua buah vektor tersebut di ruang dimensi asal, seperti berikut:

$$\mathbf{K}(\boldsymbol{x_i}, \boldsymbol{x_j}) = \emptyset(\boldsymbol{x_i}) \,.\, \emptyset(\boldsymbol{x_j}) \quad (2.31)$$

Dan prediksi pada set data dengan dimensi fitur yang baru diformulasikan dengan

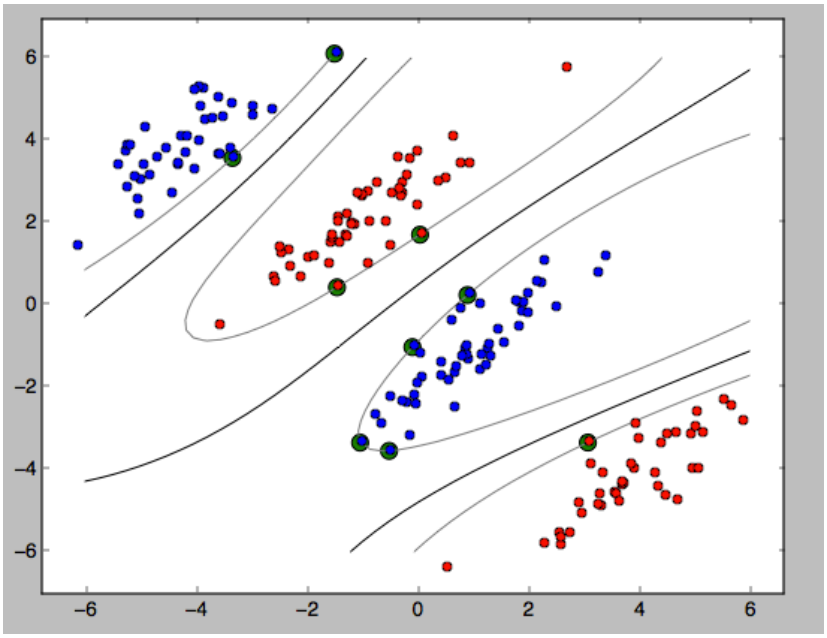
$$f(\phi(x) = sign(w \,.\, \phi(z) + b) = sign\left(\sum_{i=1}^{N} a_i y_i \phi(x_i) \,.\, \phi(z) + b\right) \quad (2.32)$$

N adalah jumlah data yang menjadi *support vector*, x_i adalah *support vector*, dan z adalah data uji yang akan diprediksi. Untuk pilihan fungsi kernel yang banyak digunakan dapat dilihat pada Tabel 2.2

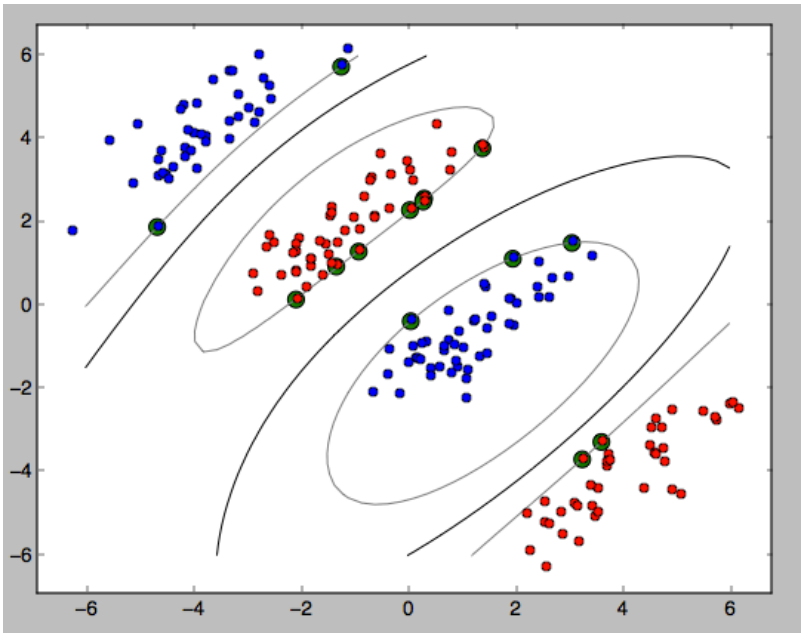
Tabel 2.2 Fungsi Kernel

Nama Kernel	**Definisi Fungsi**
Linear	K(x,y) = x.y
Polinomial	K(x,y) = $(x.y + c)^d$
Gausian RBF	exp $(-\gamma\|x - y\|^2), \gamma > 0$
Sigmoid (tangen hiperbolik)	K(x,y) = tanh(σ(x.y) + c)
Invers Multikuadratik	K(x,y) = $\frac{1}{\sqrt{\|x-y\|^2 + c^2}}$

X dan y adalah pasangan dua data dari semua bagian data latih. Parameter σ, c, d, $\gamma > 0$, merupakan konstanta.



Gambar 2.7 Diagram plot non-linear dengan kernel Gausian RBF

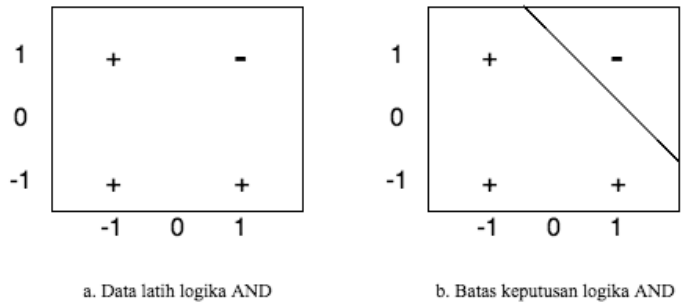


Gambar 2.8 Diagram plot non-linear dengan kernel Polinomial berderajat 2

Fungsi kernel yang sah diberikan oleh teori Mercer ([Vapnik,1995] dan [Haykin,1999]) di mana fungsi kernel harus memenuhi syarat kontinu dan pasti positif.

2.5.5 Contoh Aplikasi Klasifikasi dengan SVM

SVM pada kasus operasi AND. Dilakukan pencarian hyperplane yang tepat untuk menyelesaikan kasus operasi logika AND. Operasi logika AND merupakan contoh pemetaan data linear dengan distribusi data yang jelas dan sederhana, seperti yang ditunjukanoleh Gambar 2.9. Pertama – tama, jangkauan set data dan kelas yang digunakan dikonversi dulu agar sesuai dengan format SVM karena SVM mensyaratkan bahwa kelas yang digunakan memakai nilai -1 dan +1.



Gambar 2.9 Diagram plot data latih operasi logika AND

Tabel 2.3 merupakan contoh masalah operasi logika AND yang merupakan masalah linear sehingga tidak diperlukan fungsi kernel didalamnya.

Tabel 2.3 Operasi logika AND dengan format bipolar

x_1	x_2	Kelas(y)
1	1	1
1	-1	-1
-1	1	-1
-1	-1	-1

Dengan menggunakan Tabel 2.3, karena ada dua fitur (x1 dan x2), w juga akan memiliki dua fitur (w1 dan w2). Formulasi yang bisa digunakan adalah sebagai berikut:

Minimalkan: $\frac{1}{2}(w_1^2 + w_2^2)$

Syarat: y_i(w.x_i + b) $\geq$ 1, i = 1,2, ..., N

1. (w_1 + w_2 + b) $\geq$ 1, untuk y1 = 1; x1 = 1; x2 = 1
2. (-w_1 + w_2 - b) $\geq$ 1, untuk y1 = -1; x1 = 1; x2 = -1
3. (w_1 - w_2 - b) $\geq$ 1, untuk y1 = -1; x1 = -1; x2 = 1
4. (w_1 + w_2 - b) $\geq$ 1, untuk y1 = -1; x1 = -1; x2 = -1

Dengan menjumlahkan persamaan syarat (1) dan (2) didapatkan w2 = 1;

dengan menjumlahkan persamaan syarat (1) dan (3) didapatkan w1 = 1. Dengan menjumlahkan persamaan syarat (2) dan (3) didapatkan b = -1 sehingga persamaan hiperbolik yang didapatkan adalah

$f(x) = w_1 x_1 + w_2 x_2 + b$

$f(x) = x_1 + x_2 - 1$

2.5.6 Karakteristik *Support Vector Machine*

Karakteristik klasifikasi SVM dapat diringkas menjadi seperti berikut [1] :

1. SVM sebenarnya bisa dikatakan sebagai teknik klasifikasi yang *semi-eager-learner* karena selain memerlukan proses pelatihan, SVM juga menyimpan sebagian kecil data latih untuk digunakan kembali pada saat proses prediksi. Sebagian data yang masih disimpan ini adalah *support vector*.
2. Untuk parameter yang sama yang digunakan dalam klasifikasi, SVM memberikan model klasifikasi yang solusinya adalah *global optima*, tidak seperti ANN yang solusinya sering masuk pada wilayah *local optima*. Hal ini berarti SVM selalu memberikan model yang sama dan solusi dengan margin maksimal, sedangkan ANN memberikan model dengan nilai yang berbeda dengan margin yang tidak selalu sama.
3. SVM membutuhkan komputasi pelatihan dan prediksi yang rumit karena dimensi data yang digunakan dalam proses pelatihan dan prediksi lebih besar daripada dimensi yang sesungguhnya. Hal ini bertentangan dengan metode lain yang umumnya mengurangi dimensi untuk memberikan kinerja yang lebih cepat dan akurasi yang lebih baik.
4. Untuk set data berjumlah besar, SVM membutuhkan memori yang sangat besar untuk alokasi matriks kernel yang digunakan. Misalnya, data latih dengan ukuran 1.000 x 1.000. Metode pelatihan SVM yang membutuhkan memori besar adalah *chunking* (Vapnik, 1982) dan dekomposisi (Osuna,1997), *sedangkan sequential minimal optimization*

atau SMO (Platt,1999) dikembangkan untuk mempercepat proses pelatihan dan mengurangi memori.

5. Penggunaan matriks kernel mempunyai keutungan lain, yaitu kinerja set data dengan dimensi besar tetapi jumlah datanya sedikit akan lebih cepat karena ukuran data pada dimensi baru berkurang banyak. Misalnya, data latih berukuran 10 data 1.000 kolom fitur akan berubah menjadi matriks kernel berukuran 10x10 saja.

2.6 Multiclass Support Vector Machine (SVM)

Metode – metode klasifikasi seperti Decison Tree, Artficial Neural Network, Nearest Neighbor didesan untuk dapat melakukan klasifikasi multikelas sekaligus, tetatpi SVM tidak. SVM hanya dapat melakukan klasifikasi biner (dua kelas). Sementara masalah di dunia nyata umumnya mempunyai banyak kelas seperti pengenalan karakter, pengenalan wajah, atau diagnosis pasien, di mana data masukan terbagi menjadi lebih dari dua kelas[1].

2.6.1 Metode one-against-all

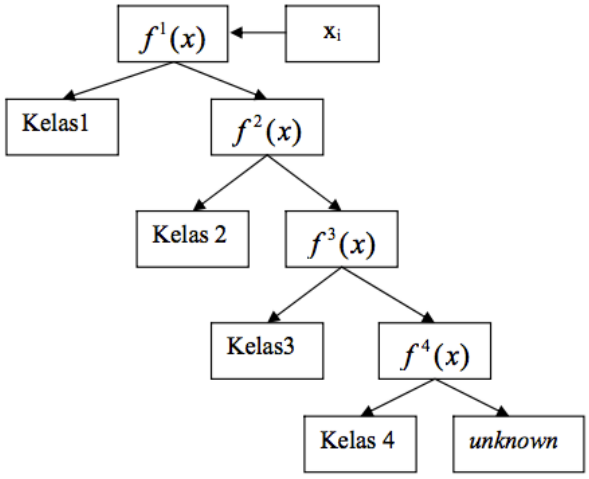
Dengan metode ini, dibangun k buah model SVM biner (k adalah jumlah kelas). Setiap model klasifikasi ke-i dilatih dengan menggunakan keseluruhan data, untuk mencari solusi permasalahan[5].

$$\begin{aligned}
&\min_{w^i,b^i,\xi^i} \frac{1}{2}\left(w^i\right)^T w^i + C\sum_t \xi_t^i \\
&s.t\ \left(w^i\right)^T \phi(x_t) + b^i \geq 1 - \xi_t^i \rightarrow y_t = i, \\
&\left(w^i\right)^T \phi(x_t) + b^i \geq -1 + \xi_t^i \rightarrow y_t \neq i, \\
&\xi_t^i \geq 0
\end{aligned} \tag{2.30}$$

Contohnya, terdapat permalasahan klasifikasi dengan 4 buah kelas. Untuk pelatihan digunakan 4 buah SVM biner.

Tabel 2.4 Contoh 4 SVM biner dengan metode *One-against-all*

$y_i = 1$	$y_i = -1$	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	$f^3(x) = (w^3)x + b^3$



Gambar 2.10 Contoh klasifikasi dengan metode *One-against-all*

2.6.2 Metode *one-against-one*

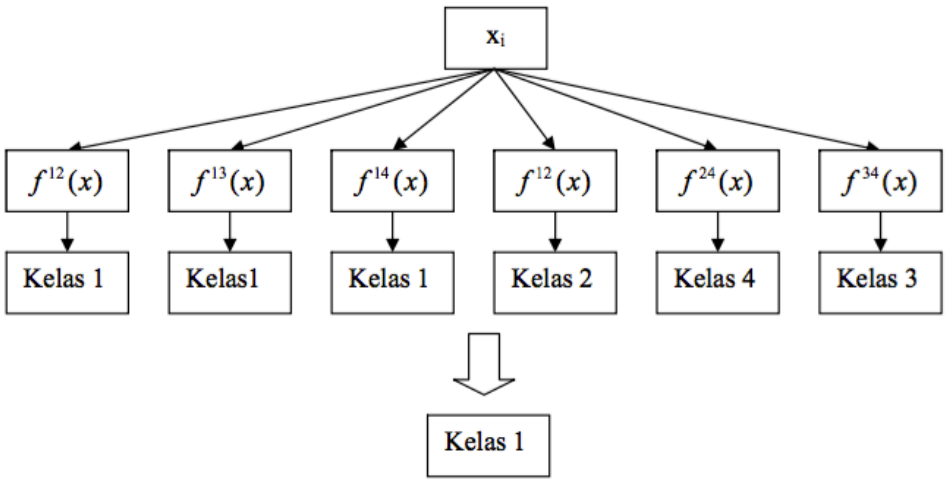
Dengan menggunakan metode ini, dibangun $\frac{k(k-1)}{2}$ buah model klasifikasi biner (k adalah jumlah kelas). Setiap model klasifikasi dilatih pada data dari dua kelas[5]. Untuk data pelatihan dari kelas ke-i dan kelas ke-j, dilakukan pencarian solusi untuk persoalan opimasi konstrain sebagai berikut :

$$\min_{w^{ij},b^{ij},\xi^{ij}} \frac{1}{2}\left(w^{ij}\right)^T w^{ij} + C\sum_t \xi_t^{ij}$$
$$s.t\ \left(w^{ij}\right)^T \phi(x_t) + b^{ij} \geq 1 - \xi_t^{ij} \rightarrow y_t = i,$$
$$\left(w^{ij}\right)^T \phi(x_t) + b^{ij} \geq -1 + \xi_t^{ij} \rightarrow y_t = j,$$
$$\xi_t^{ij} \geq 0 \qquad (2.31)$$

Terdapat beberapa metode untuk melakukan pengujian setelah keseluruhan $k(k-1)/2$ model klasifikasi selesai dibangun.

Tabel 2.5 Contoh 4 SVM biner dengan metode *One-against-one*

y_i = 1	y_i = -1	Hipotesis
Kelas 1	Kelas 2	$f^{12}(x) = (w^{12})x + b^{12}$
Kelas 1	Kelas 3	$f^{13}(x) = (w^{13})x + b^{13}$
Kelas 1	Kelas 4	$f^{14}(x) = (w^{14})x + b^{14}$
Kelas 2	Kelas 3	$f^{23}(x) = (w^{23})x + b^{23}$
Kelas 2	Kelas 4	$f^{24}(x) = (w^{24})x + b^{24}$
Kelas 3	Kelas 4	$f^{34}(x) = (w^{34})x + b^{34}$



Gambar 2.11 Contoh klasifikasi dengan metode *One-against-one*

Jika data x dimasukkan ke dalam fungsi hasil pelatihan ($f(x) = (w^T)$

$\phi(x)+ b$) dan hasilnya menyatakan menyatakan x adalah kelas i, maka suara untuk kelas i ditambah satu. Kelas dari data x akan ditentukan dari jumlah suara terbanyak. Jika terdapat dua buah kelas yang jumlah suaranya sama, maka kelas yang indeksnya lebih kecil dinyatakan sebagai kelas dari data. Jadi pada pendekatan ini terdapat $k(k-1)/2$ buah permasalahan *quadratic programming* yang masing- masing memiliki $2n / k$ variabel (n adalah jumlah data pelatihan) [5].

2.6.3 Metode *error correcting output code*

Pendekatan *error correcting output code* (ECOC) memberikan hasil prediksi yang lebih handal daripada *one-against-all*. Pendekatan ini diinspirasi dari pendekatan teori informasi untuk mengirimkan pesan melalui saluran yang ber-noise. Idenya adalah dengan menambahkan data redudan ke dalam pesan yang dikirimkan dalam bentuk *codeword* sehingga di sisi penerima dapat mendeteksi kesalahan dalam pesan yang diterima dan bahkan memulihkan pesan asli jika ada sejumlah kesalahan yang kecil. Pendekatan tersebut adalah kode *Hamming*.

Untuk masalah multikelas, setiap kelas yi dipresentasikan oleh string bit dengan panjang n yang disebut *codeword*. Selanjutnya harus dibentuk n klasifikator biner untuk memprediksi setiap bit sesuai string *codeword*. Kelas vektor uji hasil prediksi diberikan oleh *codeword* dengan jarak hammingyang paling dekat terhadap *codeword* yang dihasilkan oleh klasifikator biner. Sementara definisi jarak *hamming* adalah jumlah bit berbeda dari semua bit yang digunakan dalam representasinya.

2.7 Python

Python adalah bahasa pemrograman computer, samalayaknya seperti bahasa pemrograman lain, misalnya C, C++, Pascal, Java, PHP, Perl dan lain – lain. Sebagai bahasa pemrograman, Python tentu memiliki dialek, kosakata atau kata kunci, dan aturan tersendiri yang jelas berbeda dengan bahasa pemrograman lainnya [7].

Bahasa pemrograman Python disusun di akhir tahun 1980-an dan implementasinya baru dimulai pada Desember 1989 oleh Guido Van Rossum di Centrum Wiskunde & Information (CWI), sebuah pusat riset di bidang

matematika dan sains, Amsterdam – Belanda; sebagai suksesor atau pengganti dari bahasa pemrograman pendahulunya, bahasa pemrograman ABC, yang juga dikembangkan di CWI oleh Leo Geurts, Lambert Meertens, dan Steven Pemberton [7].

Secara umum, para programmer banyak yang menjatuhkan pilihannya ke bahasa Python karena alasan – alasan berikut [7]:

- Python memiliki konsep design yang bagus dan sederhana, yang berfokus pada kemudahan dalam penggunaan. Kode Python dirancang untuk mudah dibaca, dipelajari, digunakan ulang, dan dirawat. Selain itu, Python juga mendukung pemrograman berorientasi objek dan pemrograman fungisional.
- Python dapat meningkatkan produktivitas dan menghemat waktu bagi para programmer. Untuk memperoleh hasil program yang sama, kode Python jauh lebih sedikit dibandingkan dengan kode yang ditulis menggunakan bahasa – bahasa pemrograman lain.
- Program yang ditulis menggunakan Python dapat dijalankan di hamper semua sistem operasi (Linux, Windows, Mac, dan lain – lain), termasuk untuk perangkat – perangkat mobile
- Python bersifat gratis atau bebas (free) dan open-source, meskipun digunakan untuk kepentingan komersil.

2.7.1 Python 2.7

Sejak versi 2.7, Python 2 dan Python 3 dirilis secara simultan (paralel). Python 2.7 masih terus dipertahankan dan diperbaharui sebagai rilis terakhir dari python 2, karena banyak *programmer* Python yang enggan beralih dari Python 2 ke Python 3. Alasannya adalah karena mereka pada umumnya sudah merasa nyaman menggunakan Python 2 sehingga proses pengembangan aplikasi diharapkan jauh lebih mudah [7].

Sejak Python 2.6, komunitas Python menyertakan modul _ *future*_ ke dalam Python 2. Dengan menggunakan modul ini, para pengguna Python 2 masih dapat menikmati fitur – fitur yang ada terdapat di dalam Python 3. Python 2

dilengkapi dengan tool bernama 2to3, yang digunakan untuk mengkonversi kode Python 2 ke Python 3[7].

2.7.2 Python 3

Python 3 dikembangkan dengan tujuan jangka panjang. Fasilitas – fasilitas baru yang saat ini ditambahkan ke dalama bahasa Python hanya diimplementasikan ke dalam Python 3 dengan harapan para *programmer* yang masih menggunakan Python 2 secara bertahap bisa beralih ke Python 3.

2.8 NoSQL

NoSQL adalah konsep mengenail penyimpanan data non – relasional. NoSQL sangat berguna pada data-data yang terus - menerus berkembang, dimana data tersebut sangat kompleks sehingga sebuah database relational tidak lagi bisa mengakomodir. Salah satu bentuknya adalah ketika suatu data saling berhubungan satu sama lain, maka akan muncul proses duplikasi data. Dimana data saling memanggil ke beberapa permintaan, tambahan data baru, perubahan data, dan lain-lain dengan key yang sama. Karena faktor hubungan antar data yang sama terjadi terus-menerus, mendorong faktor redudansi data, data menjadi berlipat-lipat, dan pada akhirnya akan menyebabkan crash pada database berkonsep RDBMS. Tipe – tipe database dalam NoSQL, yaitu [10]:

- *Document Databases* : contohnya MongoDB, setiap satu object data disimpan dalam satu dokumen. Dokumen sendiri bisa terdiri dari key-value, dan value sendiri bisa berupa array atau key-value bertingkat.
- *Graph Stores* : Format penyimpanan data dalam struktur graph. Format ini sering dipakai untuk data yang saling berhubungan seperti jejaring social. Contoh database noSQL dengan format ini adalah Neo4J dan FlockDB. FlockDB dipakai oleh twitter.
- *Key – value Stores : are the simplest NoSQL databases. Every single item in the database is stored as an attribute name (or 'key'), together with its value. Examples of key-value stores are Riak and Berkeley DB. Some key-value stores, such as Redis, allow each value to have a type, such as 'integer', which adds functionality.*

- *Wide – column Stores : Column stores in NO SQL are actually hybrid row/column store unlike pure relational column databases. Although it shares the concept of column-by-column storage of columnar databases and columnar extensions to row-based databases, column stores do not store data in tables but store the data in massively distributed architectures.*

2.8.1 MongoDB

MongoDB adalah database open source berbasis dokumen (Document-Oriented Database) yang awalnya dibuat dengan bahasa C++. MongoDB sendiri sudah dikembangkan oleh 10gen sejak Oktober 2007, namun baru dipublikasikan pada Februari 2009. Selain karena performanya 4 kali lebih cepat dibandingkan MySQL serta mudah diaplikasikan, karena telah tergabung juga sebagai modul PHP.

Dalam konsep MongoDB tidak ada yang namanya tabel, kolom ataupun baris yang ada hanyalah collection (ibaratnya tabel), document (ibaratnya record). Data modelnya sendiri disebut BSON dengan struktur mirip dengan JSON. Strukturnya cukup mudah dibaca, contohnya seperti ini.

```
{
       "nama" : "Jadequeline Marsha Pricila",
       "kontak" : {
           "alamat" : "Jl. Kihapit Timur",
           "kota" : "Cimahi",
           "kodepos" : "40532",
   "telp" : "0812xxxxxx",
       }
}
```

Dengan konsep key-value yang ada pada MongoDB, setiap document otomatis memiliki index id yang unik. Hal ini membantu mempercepat proses pencarian data secara global.

MongoDB merupakan basis data yang tidak relasional, hal ini membuat MongoDB sangat cepat saat melakukan proses manipulasi data dibandingkan dengan sistem basis data relasional, selain itu MongoDB berbasis dokumen tidak memiliki struktur data yang teratur seperti tabel. MongoDB biasanya digunakan untuk membangun sebuah sistem seperti blog, forum, portal berita, social

networking, dan website yang tidak membutuhkan proses transaksi seperti sistem bank.

2.8.2 Data model

Data model adalah cara formal menggambarkan data yang digunakan dan diciptakan dalam sistem. Data di mongoDB memiliki skema yang fleksibel. Tidak seperti database SQL yang harus menentukan skema tabelnya terlebih dahulu sebelum memasukan data. *MongoDB's collection* tidak memaksakan struktur dokumen. Fleksibilitas ini memfasilitasi pemetaan dokumen ke sebuah entitas atau objek. Setiap dokumen dapat memiliki kecocokan dengan *field* data yang merepresentasi entitas, bahkan jika data memiliki variasi yang banyak. Pada penggunaanya dokumen – dokumen pada *collection* berbagi struktur yang sama[10].

Kundi dalam merancang data model pada aplikasi MongoDB berkisar pada struktur dokumen dan bagaimana aplikasi mewakili hubungan antar data. Ada dua tool yang memungkinkan aplikasi untuk merepresentasikan hubungan ini, yaitu [10]:

1. *References*

   References menyimpan hubungan antara data dengan memasukan link atau *references* untuk mengakses data yang terkait. Secara umum, ini adalah model data yang dinormalisasi. Data model ini yang digunakan pada penelitian ini.

2. *Embedded Data*

   Embedded Documents menangkap hubungan antara data dengan meyimpan data yang terkait dalam struktur dokumen tunggal. Dokumen MongoDB memungkinkan untuk struktur *embedded documents* atau array dalam dokumen. Model – model data yang *denormalized* data model memungkinkan aplikasi untuk mengambil dan memanipulasi data terkait dalam operasi database tunggal.

2.9 Pemrograman Terstruktur

Pemrograman terstruktur adalah konsep atau paradigma atau sudut pandang pemrograman yang membagi – bagi program berdasarkan fungsi – fungsi atau prosedur – prosedur yang dibutuhkan program komputer. Modul – modul (pembagian program) biasanya dibuat dengan mengelompokan fungsi – fungsi dan prosedur – prosedur yang diperlukan sebuah proses tertentu [8].

Fungsi – fungsi dan prosedur – prosedur ditulis secara sekuensial atau terurut dari atas ke bawah sesuai dengan kebergantungan antar fungsi atau prosedur (fungsi atau prosedur yang dapat dipakai oleh fungsi atau prosedur dibawahnya harus yang sudah ditulis atau dideklarasikan di atasnya) [8].

2.9.1 DFD (*Data Flow Diagram*)

Data Flow Diagram (DFD) awalnya dikembangkan oleh Chris Gane dan Trish Sarson pada tahun 1979 yang termasuk dalam *Structured Systems Analysis and Design Methodology* (SSADM) yang ditulis oleh Chris Gane dan Trish Sarson Sistem yang dikembangkan ini berbasis pada dekomposisi fungisional dari sebuah sistem [8].

Edward Yourdon dan Tom DeMarco memperkenalkan metode yang lain pada tahun 1980-an di mana mengubah persegi dengan sudut lengkung dengan lingkaran untuk menotasikan. DFD Edward Yourdon dan Tom DeMarco populer digunakan sebagai model analisis sistem perangkat lunak untuk sistem perangkat lunak yang akan diimplementasikan dengan pemrograman terstruktur [8].

DFD dapat digunakan untuk merepresentasikan sebuah sistem atau perangkat lunak pada beberapa level abstraksi. DFD dapat dibagi menjadi beberapa level yang lebih detail untuk mereprentasikan aliran informasi atau fungsi yang lebih detail [8].

2.9.1.1 Kamus Data

Kamus data dipergunakan untuk memperjelas aliran data yang digambarkan pada DFD. Kamus data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukan (*input*) dan keluaran (*output*) dapat dipahami secara umum. Kamus data dalam implementasi program

dapat menjadi parameter masukan atau keluaran dari sebuah fungsi atau prosedur [8]. Kamus data biasanya berisi:

- nama – nama dari data
- Digunakan pada - merupakan proses – proses yang terkait data
- Deskripsi – merupakan deksripsi data
- Informasi tambahan = seperti tipe data, nilai data, batas nilai data, dan komponen membentuk data

2.10 Scikit – Learn

Modul Sklearn merupakan *machine learning library* ditulis dalam python. Modul ini menyediakan banyak sekali algoritma, *datasets*, *utilities*, dan *frameworks machine learning*. Libraries numpy dan scipy yang ada dalam sklearn, biasanya dioptimasi untuk kecepatan. Ini yang membuat sklearn *fast* dan *scalable*, serta sangat berguna untuk *research users* [9].

The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes [9]:

- ***NumPy****: Base n-dimensional array package*
- ***SciPy****: Fundamental library for scientific computing*
- ***Matplotlib****: Comprehensive 2D/3D plotting*
- ***IPython****: Enhanced interactive console*
- ***Sympy****: Symbolic mathematics*
- ***Pandas****: Data structures and analysis*

2.11 NLTK

Modul *Natural Language Toolkit* (NLTK) menyediakan berbagai fungsi dan wrapper, serta corpora standar baik itu mentah atau pun pre-processed yang digunakan dalam materi pengajaran *Natural Language Processing* (NLP) [11].

Pustaka NLTK menyediakan beberapa corpora teks yang sering digunakan dalam komunitas penelitian NLP. Berikut ini, contoh corpora yang ada di dalam NLTK:

- Brown Corpus: The Brown Corpus of Standard American English dinilai sebagai corpus English pertama yang dapat dipakai dalam tugas

pemrosesan computational linguistic. Corpus ini terdiri dari sekitar sejuta kata dalam bahasa Inggris Amerika yang dicetak pada tahun 1961. Agar corpus ini dapat mewakili bahasa Inggris seoptimal mungkin, corpus ini dibagi kedalam 15 genre tulisan, misalnya fiksi, berita dan teks agama. Berikutnya versi POS-tagged dari corpus ini juga dibuat menggunakan teknik-teknik manual.

- Gutenberg Corpus: The Gutenberg Corpus terdiri dari 14 teks pilihan dari Project Gutenberg - koleksi ebook gratis terbesar. Corpus ini terdiri dari 1,7 juta kata.
- Treebank Corpus: Kumpulan teks yang sudah di-parsing dari Penn treebank.
- StopwordsCorpus: Selain kata-kata umum, ada juga kelompok kata yang disebut dengan stop words yang memiliki posisi penting dalam grammar namun tidak bisa berdiri sendiri, seperti prepositions, complementizers, dan determiners. Pustaka NLTK menyediakan Stopword Corpus yang terdiri dari 2400 kata dalam 11 bahasa berbeda (termasuk bahasa Inggris).