BAB 2
TINJAUAN PUSTAKA

2.1 *Part of Speech Tag*

Part of Speech Tag atau yang sering disingkat menjadi POS Tag merupakan bagian dari ilmu Natural Language Processing(NLP) [5]. POS Tag dapat diartikan dengan menentukan kategori morfosintaktis setiap kata dalam kalimat yang diberikan [6]. POS Tag juga dapat diartikan dengan pengklasifikasian dengan mengkategorikan setiap kata ke dalam kelas katanya.

Tabel 2.1 Contoh Pos Tag

Teks	Token	Kelas
Saya memegang tongkat	Saya	PRP
	memegang	VB
	tongkat	NN

Pada Gambar 2.1 terdapat kata yang sesuai dengan kelas katanya. “Saya” termasuk kelas kata PRP (Pronomina Persona) dimana dalam kelas tersebut diartikan sebagai kata ganti orang, contoh lainnya adalah dia, kamu, aku, mereka, kalian. Kata kedua “memegang” termasuk ke dalam kelas kata VB (Kata Kerja) yang artinya sedang melakukan suatu aktivitas atau pekerjaan. Kata ketiga sekaligus terakhir yaitu “tongkat” yang merupakan kelas kata NN (Noun/kata benda) yang tidak lain adalah berupa benda, nama orang pun termasuk ke dalam kelas kata ini.

Jumlah kelas POS Tag tiap bahasa biasanya berbeda-beda, hal ini bisa dilihat dari bentuk tata bahasa masing-masing. Tidak hanya itu, di satu negara pun masih terdapat perbedaan dalam jumlah kelas. Di indonesia sendiri memiliki 23 kelas yang telah dibuat oleh UI POS Tag dalam bentuk korpus dengan format TSV [8], untuk lebih jelasnya dapat kita lihat pada Tabel 2.1. Kelas yang digunakan sama dengan kelas pada bahasa Inggris.

Tabel 2.2 Kelas Kata

No	Tag	Deskripsi	Contoh
1	CC	Coordinating Conjunction, merupakan kata yang dipakai untuk menghubungkan satu kata dengan kata lainnya	atau, dan, tetapi
2	CD	Cardinal Number, merupakan suatu numerik atau kata yang menunjukan sebuah numerik.	ribu, tiga, miliar, 919, 0.12, sedikit, ribuan, tanggal, tahun
3	OD	Ordinal Number, merupakan kata atau nilai yang mengindikasikan posisi	ke-5, pertama, kedua
4	DT	Determiner / artikel, biasanya disimpan di depan kata benda untuk menandainya. Bisa berupa pasti atau tak tentu	sang, si, para
5	FW	Foreign Word, merupakan kata dari bahasa asing yang tidak terdapat pada kamus bahasa Indonesia	*accept, ignore.*
6	IN	Perposisi, merupakan kata penghubung dan biasanya disimpan di depan preposisinya dan menghasilkan kata preposisi	oleh, pada, di, dalam.
7	JJ	Adjective, merupakan kata dimana deskripsi, modifikasi, atau beberapa properti spesifik dari frasa noun.	biru, pendek, sebentar, jauh, sabar, manis, nasional, segitiga.
8	MD	Modal dan kata kerja bantu	harus, perlu, mesti
9	NEG	Negasi, merupakan kata yang bersifat negatif atau penolakan	jangan, tidak, belum
10	NN	Noun, merupakan kata yang menunjukan manusia, binatang, konsep, arah, berkaitan dengan waktu, dan mata uang.	kera, bawah, sekarang, rupiah
11	NNP	Proper Noun, merupakan nama spesifik dari seseorang, geografi, negara, organisasi, institusi, atau perusahaan, hari, bulan, kompetisi, dan simbol stok.	Bambang, selat sunda, Indonesia, Bank BNI, KPK, Agustus, Jumat, Idul Adha, Asean Games, Liga Primer, Harry Potter
12	NND	Classifier, merupakan kata yang mengindikasikan pada kata timbangan, membuat sebuah hal yang tadinya tidak bisa dihitung menjadi bisa dihitung.	kilogram, orang, lembar
13	PR	Demonstrative pronoun, meng-implikasikan penunjukan tempat objek.	ini, itu, sini, situ .

14	PRP	Personal pronoun, merupakan kata ganti yang mengacu pada orang. kata ganti yang termasuk ke dalam kelas ini yaitu sebagai orang pertama tunggal, orang pertama jamak, orang kedua tunggal, orang ketiga tunggal, dan orang ketiga jamak.	saya, kami, kita, kamu, dia, kalian, mereka.
15	RB	Adverb, merupakan kata keterangan yang berfungsi untuk menerangkan kata sifat, kata kerja maupun adverb lainnya.	sangat, hanya, justru, niscaya, segera
16	RP	Particle, merupakan partikel yang biasanya terdapat pada kalimat deklaratif, interogatif, ataupun imperative. Kata ini biasanya ditandai dengan partikel empati.	-kah, -lah, pun
17	SC	Subordinating conjunction atau bisa juga disebut dengan subordinator, merupakan penghubung antara 2 atau lebih klausa yang biasanya dibagi menjadi klausa utama dan klausa pendukung(subordinat).	maka, yang, tanpa, semoga, dengan, bahwa.
18	SYM	Symbol, termasuk simbol uang, simbol matematik yang biasanya dilabelkan dengan SYM.	IDR, +, %, @
19	UH	Interjection, merupakan kata seruan dengan ciri mengungkapkan perasaan	Huh, hey, ayo
20	VB	Verbs, merupakan kata kerja atau kata yang menunjukan suatu aktivitas.	makan, belajar, mencuci.
21	WH	Question word, merupakan kata-kata yang bersifat tanya.	bagaimana, berapa, siapa, dimana.
22	X	Unknown, merupakan kata yang tidak termasuk dalam kelas kata lainnya seperti typo, kata yang belum diketahui.	kesimpalan
23	Z	Punctuation, merupakan tanda baca atau symbol yang tidak termasuk kata atau frasa dalam suatu bahasa.	?, “.”, “,”, !.

2.2 Blok Diagram

Blok diagram merupakan suatu cara untuk merepresentasikan sistem yang terdiri dari input, proses, output. Dalam diagram ini, proses akan digambarkan dengan blok, sedangkan input dan output digambarkan dengan dengan panah [13]. Berikut struktur diagram blok secara umum.

Gambar 2.1 Blok Diagram Secara Umum

2.3 Data Flow Diagram

Dalam menggambarkan arus data sistem terstruktur atau sequential diperlukan Data Flow Diagram(DFD). DFD juga bisa disebut dengan DAD atau Diagram Arus Data. Dengan diagram ini memperlihatkan gambaran tentang input-proses-output yang ada pada sistem. Ada dua tipe DFD yang biasanya digunakan, yang pertama penggambaran berdasarkan Gane, Sarson dan yang kedua berdasarkan Yourdon, De Marco. Dalam penelitian ini menggunakan tipe DFD yang dibuat oleh Yourdon, De Marco. Ciri dari DFD tersebut yaitu proses yang digambarkan dengan lingkaran dan garis sejajar yang dianggap sebagai *data store*. DFD ini memiliki empat komponen diantaranya proses, arus, entitas, dan *data store* [14].

2.4 Diagram Konteks

Diagram Konteks merupakan model diagram yang terdiri dari satu proses saja. Diagram ini masih termasuk ke dalam DFD dan merupakan top level dari DFD. Jadi sebelum membuat DFD level-level bawah maka harus dibuat dulu Diagram Konteks untuk menggambarkan sistem secara garis besarnya. Walaupun hanya memiliki satu proses saja, Diagram konteks memiliki tiga komponen yaitu proses, arus, dan entitas [14].

2.5 Ekstraksi Fitur

Fitur bisa diartikan sebagai ciri/informasi, jadi proses ekstraksi fitur adalah proses mengambil ciri/informasi yang terkandung dalam sebuah data [15]. Data

yang dimaksud pada penelitian ini adalah token-token. Fitur-fitur yang digunakan pada penelitian ini ada 14 fitur [5] [6] [10] diantaranya.

Tabel 2.3 Fitur

Fitur	Ketentuan	Contoh
Caps	bernilai *True* jika awal huruf dari token yang diperiksa berupa kapital	***G**ates*, ***D**ia*
In_Cap	Bernilai True jika token yang diperiksa mengandung kapital kecuali awal huruf	*i**P**hone*
All_Cap	Bernilai True jika semua huruf dari token yang diperiksa berupa kapital	***SBY***
All_Low	Bernilai True jika semua huruf dari token yang diperiksa berupa huruf kecil	***Selalu, bekerja***
Num	Bernilai True jika token yang diperiksa merupakan bilangan numerik.	***1, 20***
Hyp	Bernilai True jika token yang diperiksa mengandung simbol (“-“).	*Bolak-balik, kejar-kejaran*
Me-	Bernilai True jika token yang diperiksa mengandung imbuhan “me“	***Me**lakukan, **me**melihara*
Pe-	Bernilai True jika token yang diperiksa mengandung imbuhan “pe”	***Pe**laksanaan, **pe**meriksaan*
Ke-	Bernilai True jika token yang diperiksa mengandung imbuhan “ke”	***Ke**laparan, **ke**keliruan*
Se-	Bernilai True jika token yang diperiksa mengandung imbuhan “se”	***Se**ekor*
Be-	Bernilai True jika token yang diperiksa mengandung imbuhan “be”	***Ber**main, **ber**napas*
di-	Bernilai True jika token yang diperiksa mengandung imbuhan “di”	***Di**diamkan, **di**rendam*
-an	Bernilai True jika token yang diperiksa mengandung imbuhan “an”	*Pengaku**an,** kekeliru**an***
-kan	Bernilai True jika token yang diperiksa mengandung imbuhan kan	*Panas**kan,** kobar**kan***

2.6 *Support Vector Machine*

Support Vector Machine (SVM) adalah sebuah machine learning yang pertama kali diperkenalkan oleh Vapnik. SVM merupakan supervised machine learning yang dapat menyelesaikan berbagai masalah seperti kategori teks, tulisan tangan, digit recognition, tone recognition, klasifikasi gambar dan deteksi objek, dan klasifikasi data [16]. Machine learning menjadi salah satu andalan di bidang teknologi informasi dalam memecahkan masalah dan analisis terkait data yang semakin banyak karena dipercaya bahwa sistem analisis yang pintar akan diperlukan untuk kemajuan teknologi. Dengan kecepatan tinggi dan akurasi yang bagus dapat mempersingkat waktu dan juga mencapai kualitas yang diperlukan serta menghindari human error.

2.6.1 Teori SVM

Dalam metode SVM, poin utamanya yaitu untuk mengoptimalkan yang namanya *hyperplane*. *Hyperplane* digunakan sebagai batas yang memisahkan *support vector* kelas satu dengan *support vector* kelas lainnya. Dengan mengoptimalkan *support vector* khususnya *support vector* yang berdekatan antara kelas satu dengan kelas lainnya dijadikan sebagai tolak ukur untuk batas klasifikasi agar *hyperplane* yang akan dibuat menjadi optimal. *Vector* ini berasal dari dataset yang sudah diubah menjadi nilai *vector* melalui *vectorization* setelah proses ekstraksi fitur dan dijadikan sebagai *support vector*. Sebagai contoh pada dataset *training* terdiri dari x dan y dalam bentuk $\{(x_1, y_1), \dots, (x_n, y_n)\}$ dimana x disebut dengan *vector* dan y adalah label kelasnya [17].

Dalam pembuatan *hyperplane* dibutuhkan beberapa parameter yaitu w, x, dan b. Hal itu dapat ditulis ke dalam persamaan 2.1.

$$\boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b} = \mathbf{0} \qquad \textbf{(2. 1)}$$

Hyperplane akan membagi antara satu kelas dengan kelas lainnya. Kelas tersebut diberi nilai 1 dan -1 seperti terlihat pada persamaan 2.2 dan persamaan 2.3.

$$\boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b} \geq +\mathbf{1} \qquad \textbf{(2. 2)}$$

$$\boldsymbol{w}.\boldsymbol{x_i} + \boldsymbol{b} \leq -\mathbf{1} \qquad \textbf{(2. 3)}$$

Tidak hanya itu, svm juga perlu untuk memaksimalkan jarak margin antara kelas satu dan yang lainnya. Hal ini bisa dilakukan dengan fungsi quadratic programming(QP). Dalam perkembangannya, metode QP ini diubah ke dalam fungsi Lagrangian pada persamaan 2.4.

$$(\boldsymbol{w}, \boldsymbol{b}, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{n} \boldsymbol{\alpha_i y_i}(\boldsymbol{w^T x_i} + \boldsymbol{b}) \pm \sum_{i=1}^{n} \boldsymbol{\alpha_i} \quad \textbf{(2. 4)}$$

Dengan meminimalkan L yang dinyatakan pada persamaan 2.5 akan menghasilkan persamaan 2.7.

$$\boldsymbol{Min\ L}(\boldsymbol{w}, \boldsymbol{b}, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{n} \boldsymbol{\alpha_i y_i}(\boldsymbol{w^T x_i} + \boldsymbol{b}) + \sum_{i=1}^{n} \boldsymbol{\alpha_{i=0}} \quad \textbf{(2. 5)}$$

$$\boldsymbol{w} - \sum_{i=1}^{n} \boldsymbol{\alpha_i y_i} = \boldsymbol{0} \quad \textbf{(2. 6)}$$

$$\sum_{i=1}^{n} \boldsymbol{\alpha_i y_i} = \boldsymbol{0} \quad \textbf{(2. 7)}$$

Sedangkan jika dilakukan maksimalisasi L terhadap ai, maka akan didapatkan model persamaan 2.9.

$$\boldsymbol{Min}\ \sum_{i=1}^{n} \boldsymbol{\alpha_{i=0}} - \frac{1}{2}\sum_{i=1,j=1}^{n} \boldsymbol{\alpha_i y_i y_j^T x_i x^T} \quad \textbf{(2. 8)}$$

$$\sum_{i=1}^{n} \boldsymbol{\alpha_i y_i} = \boldsymbol{0}\,, \boldsymbol{\alpha_i} > \boldsymbol{0}(\boldsymbol{i}, \boldsymbol{j} = \boldsymbol{1}, \dots, \boldsymbol{n}) \quad \textbf{(2. 9)}$$

Dalam mencari α_i, masukan dahulu nilai *vector(support vector)* dari setiap data input ke dalam kernel trick phi (∅). Kernel trick phi dapat dilihat pada persamaan 2. .

$$\emptyset \begin{bmatrix} k_i \\ l_i \end{bmatrix} \begin{cases} \sqrt{\boldsymbol{k_n^2 + l_n^2}} > \boldsymbol{2}, \boldsymbol{maka} \begin{bmatrix} \sqrt{k_n^2 + l_n^2} - k_i + [k_i - l_i] \\ \sqrt{k_n^2 + l_n^2} - l_i + [k_i - l_i] \end{bmatrix} \\ \sqrt{\boldsymbol{k_n^2 + l_n^2}} > \boldsymbol{2}, \boldsymbol{maka} \begin{bmatrix} k_i \\ l_i \end{bmatrix} \end{cases} \quad \textbf{(2. 10)}$$

Dimana**:**

k = hasil kernelisasi bobot fitur

l = hasil kernelisasi kelas

i = nilai dari 1 hingga ke-n

n = jumlah data

Dimana k dan l didapat dari Persamaan 2.11 dan Persamaan 2.12. Persamaan 2.11 merupakan fungsi kernel(linear) untuk bobot token, sedangkan

Persamaan 2.12 menggunakan fungsi pelabelan *multiclass* terlebih dahulu dan kemudian kernel(linear).

$$\sum_{i=1}^{n} \boldsymbol{x_i x_j} = \boldsymbol{x_i^T x_j (i, j = 1, \dots, n)} \quad \textbf{(2. 11)}$$

$$\sum_{i=1}^{n} \boldsymbol{y_i y_j} = \boldsymbol{y_i^T y_j (i, j = 1, \dots, n)} \quad \textbf{(2. 12)}$$

Dimana:

$x =$ Bobot *vector* fitur

$y =$ Label kelas

$i =$ dari 1 hingga ke-n

$n =$ jumlah data

Sebelum masuk ke persamaan 2.14, Cari parameter a_i terlebih dahulu pada persamaan 2.13. Jika parameter sudah ketemu maka masukan parameter tersebut ke persamaan 2.14 untuk mencari nilai a_i.

$$\sum_{i=1,j=1}^{n} \boldsymbol{\alpha_i s_i^T s_j} \quad \textbf{(2. 13)}$$

$$\sum_{i=1,j=1}^{n} \boldsymbol{\alpha_i s_i^T s_j} = \boldsymbol{y_i} \quad \textbf{(2. 14)}$$

Dimana:

α = alpha

s = *support vector*

y = label kelas

Dalam memenuhi fungsi 2.16, masih diperlukan nilai w yang bisa dicari melalui persamaan 2.15. persamaan 2.16 merupakan model yang akan digunakan untuk tahap *testing*.

$$\boldsymbol{w} = \sum_{i=1,j=1}^{n} \boldsymbol{\alpha_i s_i} \quad \textbf{(2. 15)}$$

$$\boldsymbol{y = wx + b} \quad \textbf{(2. 16)}$$

Dimana:

x = data input(*vector*)

y = kelas

w = weight

b = bias

a = alpha(lagrange multiplier)

2.6.2 Kernel

Kernel pada metode *Support Vector Machine* adalah pemisah antara kelas satu dengan kelas lainnya. Ada beberapa kernel untuk *support vector machine* diantaranya Linear, Polinomial, dan Radial Basis Function(RBF). Dalam penelitian-penelitian sebelumnya, kernel yang sering digunakan adalah Linear dan Radial Basis Function(RBF). [18]

1. Linear

Linear, seperti namanya yang berarti garis lurus. Kernel linear menggunakan garis lurus sebagai pembatas/*hyperplane* antar kelas.

$$\mathbf{k}\left(\boldsymbol{x_i}, \boldsymbol{x_j}\right) = \boldsymbol{x_i^T x_j} \qquad \textbf{(2. 17)}$$

2. RBF

Dalam penyelesaiannya RBF membutuhkan parameter gamma dan C. Gamma berfungsi sebagai batas keputusan dan wilayah keputusan, sebagai contoh jika gamma bernilai kecil maka batas keputusan akan kecil namun wilayah keputusan akan menjadi luas dan begitupun sebaliknya. C berfungsi sebagai penalti terhadap kesalahan dalam klasifikasi.

$$\mathbf{exp}(-\boldsymbol{\gamma}||\boldsymbol{x_i} - \boldsymbol{x}||^{\mathbf{2}}), \boldsymbol{\gamma} > \mathbf{0} \qquad \textbf{(2. 18)}$$

3. Polinomial

Kernel Polinomial memiliki dua parameter berbeda dari kernel lain. Parameter r adalah parameter bebas yang jika diisi r=0 maka disebut homogen. Sedangkan parameter d adalah derajat/quadrat yang umumnya diisi dengan d=2

$$\mathbf{k}(\boldsymbol{x_i}, \boldsymbol{x}) = \left(\boldsymbol{y}.\ \boldsymbol{x_i^T}\ \boldsymbol{x}\ + \boldsymbol{r}\right)^{\boldsymbol{d}} \qquad \textbf{(2. 19)}$$

Pada Penelitian ini, kernel yang digunakan adalah kernel linear karena berdasarkan kesimpulan dari penelitian sebelumnya [19] bahwa akurasi kernel linear dibandingkan dengan kernel lainnya bisa jadi sama atau lebih kecil, namun disisi lain kernel linear merupakan yang paling baik dalam hal kemudahan dan waktu komputasi. Kernel ini terdapat pada persamaan 2.17.

2.6.3 Multi Class

Terdapat dua teknik multi class yang sering digunakan pada SVM yaitu *One Versus One* (OVO) dan *One Versus All* (OVA). OVA yang dimaksud yaitu membandingkan satu dengan semua selain dirinya yang dianggap menjadi satu kesatuan. *Multiclass* ini digunakan karena sejatinya SVM adalah machine learning yang hanya mengklasifikan dua kelas saja secara linear.

Misalnya, terdapat 4 buah kelas dalam permasalahan tersebut maka model yang terbentuk mengacu pada Tabel 2.4 [20].

Tabel 2.4 SVM biner metode *One Versus All*

$h_i = 1$	$h_1 = -1$	Hipotesis
Kelas 1	Bukan Kelas 1	$f^1(g) = (w^1)g + b^1$
Kelas 2	Bukan Kelas 2	$f^2(g) = (w^2)g + b^2$
Kelas 3	Bukan Kelas 3	$f^3(g) = (w^3)g + b^3$
Kelas 4	Bukan Kelas 4	$f^4(g) = (w^4)g + b^4$

Pada penelitian-penelitian sebelumnya, diketahui bahwa akurasi OVA lebih baik dibanding OVO walaupun dari segi kecepatan OVO lebih cepat [9]. Maka dari itu, pada penelitian yang akan dilakukan mengenai penerapan SVM terhadap POS Tag bahasa Indonesia ini menggunakan OVA. Dengan metode tersebut diharapkan dapat memberikan akurasi yang tinggi.

Dalam menerapkan metode OVA akan dibangun *z* buah model SVM biner. *z* disini adalah jumlah kelas. Dalam mengklasifikasikan hal tersebut dapat dilihat pada persamaan berikut [21].

$$\boldsymbol{Kelas\ g = arg\ \max_{r=1..z}((w^{(r)})^T . \varphi(g) + b^{(r)})} \qquad \textbf{(2. 20)}$$

2.6.4 *Training*

Pada proses *training* SVM langkah pertama adalah mengubah bobot fitur hasil ekstraksi fitur menjadi format yang bisa diterima oleh SVM yaitu *vector*. Langkah selanjutnya adalah memberi label pada kelas dengan 1 atau -1. Setelah itu masukan pada kernel, pada penelitian ini kernel yang digunakan adalah kernel

linear. Persamaan diambil pada persamaan 2.11 untuk bobot fitur dan persamaan 2.12 untuk label kelas.

$$\sum_{i=1}^{n} \boldsymbol{x_i x_j} = \boldsymbol{x_i^T x_j (i, j = 1, \dots, n)} \quad \textbf{(2. 21)}$$

$$\sum_{i=1}^{n} \boldsymbol{y_i y_j} = \boldsymbol{y_i^T y_j (i, j = 1, \dots, n)} \quad \textbf{(2. 22)}$$

Lakukan langkah tersebut dari *i* sama dengan 1 hingga ke-*n*, begitupun dengan *j*. *n* disini adalah jumlah dari data. Setelah itu buat matriks dengan menggunakan persamaan berikut.

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{x_i^T x_j} & \cdots & \boldsymbol{x_i^T x_{jn}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{x_{in}^T x_j} & \cdots & \boldsymbol{x_{in}^T x_{jn}} \end{bmatrix} \quad \textbf{(2. 23)}$$

$$\boldsymbol{D} = \begin{bmatrix} \boldsymbol{y_1^T y_1} & \cdots & \boldsymbol{y_1^T y_{19}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{y_{19}^T y_1} & \cdots & \boldsymbol{y_{19}^T y_{19}} \end{bmatrix} \quad \textbf{(2. 24)}$$

Pada persamaan tersebut, *i* dimulai dari 1 hingga ke-*in*, sedangkan *j* dimulai dari 1 hingga ke-*jn*. *in* dan *jn* merupakan jumlah dari data. setelah selesai tambahkan tiap baris matriksnya sehingga didapat nilai *k* dan *l*.

$$\boldsymbol{k_i} = \boldsymbol{x_i^T x_1 + \cdots + x_i^T x_j} \quad \textbf{(2. 25)}$$

$$\boldsymbol{l_i} = \boldsymbol{y_i^T y_1 + \cdots + y_i^T y_j} \quad \textbf{(2. 26)}$$

Lakukan hal tersebut dari *i* sama dengan 1 hingga ke-*n*. Selanjutnya lakukan persamaan 2.27.

$$\boldsymbol{\varphi}\begin{bmatrix} \boldsymbol{k_i} \\ \boldsymbol{l_i} \end{bmatrix} = \begin{cases} \sqrt{\boldsymbol{k_n}^2 + \boldsymbol{l_n}^2} > \boldsymbol{2, maka} \begin{bmatrix} \sqrt{\boldsymbol{k_n}^2 + \boldsymbol{l_n}^2} - \boldsymbol{k_i} + |\boldsymbol{k_i} - \boldsymbol{l_i}| \\ \sqrt{\boldsymbol{k_n}^2 + \boldsymbol{l_n}^2} - \boldsymbol{l_i} + |\boldsymbol{k_i} - \boldsymbol{l_i}| \end{bmatrix} \\ \sqrt{\boldsymbol{k_n}^2 + \boldsymbol{l_n}^2} \leq \boldsymbol{2, maka} \begin{bmatrix} \boldsymbol{k_i} \\ \boldsymbol{l_i} \end{bmatrix} \end{cases} \quad \textbf{(2. 27)}$$

Perhitungan dimulai dengan mengecek apakah hasilnya akan bernilai lebih dari 2 atau bisa jadi sama dengan maupun kurang dari 2. Jika hasilnya lebih dari 2 maka rumus yang digunakan adalah yang berada diatas, namun jika tidak maka hasilnya sama dengan $\begin{bmatrix} p_i \\ q_i \end{bmatrix}$ [22].

Persamaan tersebut akan menghasilkan nilai *support vector*. Dalam mencari nilai α, *support vector* ditambahkan nilai bias 1 agar tegak lurus sempurna. Lalu lakukan persamaan 2.28.

$$\sum_{i=1,j=1}^{n} \alpha_i s_i^T s_j = l_i \quad \textbf{(2. 28)}$$

Lakukan langkah tersebut dari *i* sama dengan 1 hingga ke-*n*, begitupun dengan *j*. Jika nilai α telah didapatkan, maka langkah selanjutnya adalah mencari nilai *w* dan *b* yang baru sebagai *hyperplane*.

$$w = \sum_{i=1}^{n} \alpha_i s_i \textbf{dimana } \alpha_i \geq 0 \quad \textbf{(2. 29)}$$

2.6.5 *Testing*

Sama halnya dengan proses *training*, bobot harus diubah menjadi *vector* baik, sedangkan untuk kelas diberi label. Pada proses pelabelan kelas untuk data *testing* dianggap sama dengan 0, sedangkan label untuk kelas data yang lainnya sama seperti pada proses *training* [17]. Setelah itu cari variabel yang dibutuhkan dalam perhitungan kernel pada persamaan 2.30 dan persamaan 2.31. Lakukan perhitungan berikut dengan *i* dari 1 hingga ke-*n*, begitupun dengan *j*. Nilai *n* adalah jumlah data.

$$\sum_{i=1}^{n} g_i g_j = g_i^T g_j (i, j = 1, \dots, n) \quad \textbf{(2. 30)}$$

$$\sum_{i=1}^{n} h_i h_j = h_i^T h_j (i, j = 1, \dots, n) \quad \textbf{(2. 31)}$$

Setelah itu petakan ke dalam matriks. Persamaan 2.32 untuk *vector* bobot *testing* dan persamaan 2.33 untuk kelas *testing*.

$$E = \begin{bmatrix} g_1^T g_1 & \cdots & g_1^T g_{20} \\ \vdots & \ddots & \vdots \\ g_{20}^T g_1 & \cdots & g_{20}^T g_{20} \end{bmatrix} \quad \textbf{(2. 32)}$$

$$F = \begin{bmatrix} h_1^T h_1 & \cdots & h_1^T h_{20} \\ \vdots & \ddots & \vdots \\ h_{20}^T h_1 & \cdots & h_{20}^T h_{20} \end{bmatrix} \quad \textbf{(2. 33)}$$

Lalu masukan ke dalam kernel dengan persamaan 2.34 untuk *vector* dan persamaan 2.35 untuk kelas.

$$p_i = g_i^T g_j + \cdots + g_i^T g_j \quad \textbf{(2. 34)}$$

$$q_i = h_i^T h_j + \cdots + h_i^T h_j \quad \textbf{(2. 35)}$$

Setelah didapat nilai kernel p_i dan $q_i$maka cari support *vector*. Berikut persamaan untuk mencari support *vector testing*.

$$\boldsymbol{\varphi}\begin{bmatrix}\boldsymbol{p_i}\\ \boldsymbol{q_i}\end{bmatrix} = \begin{cases} \sqrt{\boldsymbol{p_n}^2 + \boldsymbol{q_n}^2} > \boldsymbol{2},\ \boldsymbol{maka} \begin{bmatrix}\sqrt{\boldsymbol{p_n}^2 + \boldsymbol{q_n}^2} - \boldsymbol{p_i} + |\boldsymbol{p_i} - \boldsymbol{q_i}|\\ \sqrt{\boldsymbol{p_n}^2 + \boldsymbol{q_n}^2} - \boldsymbol{q_i} + |\boldsymbol{p_i} - \boldsymbol{q_i}|\end{bmatrix} \\ \sqrt{\boldsymbol{p_n}^2 + \boldsymbol{q_n}^2} \leq \boldsymbol{2},\ \boldsymbol{maka} \begin{bmatrix}\boldsymbol{p_i}\\ \boldsymbol{q_i}\end{bmatrix} \end{cases} \quad \textbf{(2. 36)}$$

Tahap terakhir dari proses *testing* yaitu masukan *support vector* data *testing* ke dalam persamaan 2.20. Hasil nilai terbesar pada perhitungan tersebut merupakan kelas prediksinya.

2.7 Nilai Akurasi

Nilai akurasi dihitung untuk menentukan tingkat keberhasilan klasifikasi atau seberapa baik model yang telah dibuat. Berikut persamaan untuk mencari nilai akurasi [23].

$$Akurasi = \frac{Jumlah\ kelas\ prediksi\ benar}{Total\ token\ testing} * 100\% \quad \textbf{(2. 37)}$$

2.8 Python

Python adalah bahasa pemrograman yang menyediakan struktur tingkat tinggi seperti daftar dan susunan asosiatif(kamus), pengetikan dinamis dan pengikatan dinamis [24]. Bahasa ini dirancang oleh Guidon Van Rossum pada tahun 1990, memiliki sintaks yang sederhana dan dapat dijalankan secara praktis. Programnya dikompilasi dengan cara menafsirkan kode byte setelah menginterpreter ke dalam platform independen [24].

2.9 Sklearn

Sklearn adalah modul terintegrasi untuk python yang didalamnya memuat librari machine learning baik supervised dan unsupervised. [24]. Sklearn bertujuan untuk memudahkan pengguna dalam mempelajari dan mengimplementasikan machine learning. Beberapa machine learning yang terdapat di dalam sklearn diantaranya CRF, Tree, SVM, dan Naive Bayes. Maka dari itu, librari ini juga bisa bermanfaat dalam analisis data.

2.10 NLTK

NLTK yang merupakan kependekan dari The Natural Language Toolkit adalah librari yang sangat populer dan sering digunakan dalam penelitian dan pembelajaran. Librari ini memiliki banyak kegunaan diantaranya dataset, tutorial dan latihan, dan statistika pada *natural language processing* [25]. Dalam penelitian ini NLTK digunakan dalam proses *preprocessing* yaitu untuk membagi teks kedalam token-token atau sering disebut dengan tokenisasi.