BAB II
KAJIAN LITERATUR

2.1 Pendahuluan

Bab ini akan menguraikan teori yang melandasi penelitian. Uraian teori tersebut terdiri dari deskripsi mengenai pertanyaan, klasifikasi teks, seleksi fitur, algoritma yang digunakan dalam penelitian yakni seleksi fitur *Information Gain*, *Chi-Square* dan *Mutual Information*, beberapa penelitian yang relevan, serta kesimpulan.

2.2 Landasan Teori

2.2.1 Pertanyaan

Pertanyaan merupakan suatu cara yang dilakukan seseorang dalam rangka memperoleh informasi atau mengkonfirmasi suatu hal melalui ekspresi yang disampaikan lewat suatu kalimat tanya baik secara lisan maupun tulisan (Sudin et al., 2019). Sebagai suatu frasa, kalimat tanya khususnya kalimat tanya berbahasa Indonesia tentu memiliki ciri tersendiri diantaranya, yakni :

1. Kalimat tanya selalu diakhiri dengan simbol berupa tanda baca tanya atau ‘?’.
2. Kalimat tanya biasanya diawali dengan kata tanya berupa 5W+1H diantaranya apa, kapan, dimana, mengapa, siapa dan bagaimana.
3. Kalimat tanya sering menggunakan imbuhan –kah pada kata tanya seperti siapakah, dimanakah dan sebagainya.
4. Kalimat tanya yang tidak diawali dengan kata tanya dapat menambahkan imbuhan –kan diakhir kalimat seperti “Kamu belum mandi, iya kan ?”.

5. Kalimat tanya tentunya membutuhkan tanggapan atau jawaban yang jelas terkait informasi yang diberikan.

Berdasarkan jawaban atau informasi yang diberikan, kalimat tanya juga dapat diklasifikasikan menjadi beberapa jenis pertanyaan, diantaranya: :

a) Pertanyaan *Factoid*

Pertanyaan *factoid* merupakan suatu pertanyaan yang hanya membutuhkan jawaban berupa fakta singkat dan ringkas seperti entitas, angka dan nama. Contohnya pertanyaan *factoid*, ialah:

1. "kucing jenis apa yang berbulu tebal ?"
2. "dimana kamu membeli baju tersebut ?"

b) Pertanyaan *Non-Factoid*

Pertanyaan *non-factoid* merupakan suatu pertanyaan yang membutuhkan jawaban berisi fakta yang cukup panjang disertai dengan penjelasan terkait suatu hal. Contoh dari pertanyaan *non-factoid*, seperti:

1. "bagaimana kau bisa tiba disini begitu cepat ?"
2. "lantas mengapa kau tak menyampaikan yang sebenarnya ?"

c) Pertanyaan *Others*

Pertanyaan *others* merupakan suatu pertanyaan yang membutuhkan jawaban diluar dari pertanyaan *factoid* dan *non-factoid* seperti pertanyaan dengan jawaban bertipe *yes-no*, list, dan opini. Contoh dari pertanyaan *others*, ialah:

1. "tahukah kamu dimana fakultas teknik ?"
2. "apa saja barang yang ingin kau jual kepadaku ?"

Dalam menentukan kategori dari suatu kalimat tanya berdasarkan jawaban atas informasi yang telah diuraikan pada pembahasan sebelumnya, diperlukan adanya analisa lebih lanjut dikarenakan kata yang mewakili kalimat tanya seperti “apa” dapat menghasilkan ambiguitas yang cukup tinggi pada hasil kategori. Seperti kalimat tanya “apa nama ibu kota Indonesia ?” dan “apakah hari ini kamu sudah makan nasi ?” dapat menghasilkan jawaban dengan kategori yang berbeda.

2.2.2 Klasifikasi Teks

Klasifikasi teks merupakan bagian penelitian dalam bidang *Information Retrieval* yang mengembangkan suatu cara dalam melakukan pengelompokkan dan penentuan suatu dokumen teks kedalam satu kelas atau lebih secara otomatis (Februariyanti & Zuliarso, 2012). Klasifikasi teks merupakan suatu pekerjaan yang ditujukan untuk menentukan apakah suatu teks dokumen termasuk dalam kategori tertentu (Suharno et al., 2017).

Pra-pengolahan (*preprocessing*) adalah rangkaian awal yang dilakukan dalam pemrosesan teks dengan tujuan untuk mengolah, mengatur dan menggali data yang digunakan agar data tersebut menjadi lebih tersetruktur sehingga *noise* yang ada pada dataset dapat berkurang. Hal ini penting karena akan mempengaruhi hasil akhir dari proses klasifikasi nantinya. Rangkaian tahapan *preprocessing* juga terdiri dari beberapa tahapan seperti *Case Folding* yang berfungsi untuk mengkonversi teks agar menjadi bentuk yang selaras, *Tokenizing* yang digunakan untuk memecah kalimat yang ada teks menjadi satu sukuan kata, *Filtering* yang dapat mereduksi variable yang tidak mempunyai korelasi dengan dokumen yang

digunakan, dan *Stemming* yang ditujukan untuk memperkecil jumlah indeks yang berbeda pada suatu dokumen dengan mengembalikan suatu kata ke kata dasarnya (Rahman et al., 2021).

Perhitungan pembobotan Fitur TF- IDF dapat menggunakan rumus yang ada pada persamaan II-1 dan II-2.

$$W_{ij} = TF(i, j)x\ IDF \qquad \text{(II-1)}$$

Rumus mencari nilai *Inverse Document Frequency* (IDF) menggunakan persamaan II-2.

$$IDF\ = \log\ (\frac{N}{DF(j)}) + 1 \qquad \text{(II-2)}$$

Keterangan :

- N : Banyaknya dokumen
- W*ij* : Bobot nilai
- TF(*i, j*) : Jumlah kemunculan setiap kata *i* dalam sebuah dokumen *j*.
- DF(*i*) : Jumlah dokumen yang mengandung kata *i*.

Setelah dilakukan proses pembobotan fitur maka dapat dilakukan proses klasifikasi yang bertujuan untuk membagi setiap entitas kedalam kelas yang telah ditentukan. Pengklasifikasian sendiri terbagi menjadi dua jenis yaitu *supervised* dan *unsupervised*. Perbedaan keduanya terletak di target kelas, apabila target kelasnya

telah ditentukan maka termasuk klasifikasi *supervised*. Jika target kelas belum ditentukan pada dataset maka termasuk klasifikasi *unsupervised* (Hanati & Sari, 2021).

Untuk mendapatkan hasil kinerja yang lebih maksimal, dapat ditambahkan proses seleksi fitur setelah dilakukan tahapan pra-pengolahan teks yang berguna untuk menemukan fitur terbaik sebelum dilakukan proses pembobotan fitur dan klasifikasi teks (Ridok & Latifah, 2015).

2.2.3 Seleksi Fitur

Seleksi fitur ialah salah satu dari serangkaian tahap preprocessing ketika tahapan *stemming* dan penggunaan *stopword list* masih dirasa kurang dalam mereduksi dimensi (Suharno et al., 2017). Cara kerja seleksi fitur ialah dengan menentukan fitur yang relevan dan melakukan pembobotan pada setiap fitur. Seleksi fitur sendiri bertujuan untuk meningkatkan efisiensi dan efektifitas hasil kinerja dari algortima klasifikasi dengan mengurangi fitur yang tidak relevan dan banyaknya dimensi data (Buani, 2021).

Seleksi fitur terbagi menjadi dua jenis metode yaitu yaitu *filter* dan *wrapper*. Metode *wrapper* melakukan interaksi dengan algoritma klasifikasi dalam menentukan kegunaan suatu fitur sehingga menghasilkan *sub* fitur dengan kinerja yang lebih baik. Namun, cenderung lebih lambat jika dibandingkan dengan metode *filter* (Tanti et al., 2021). Metode filter sendiri memerlukan *confusion matrix* untuk melakukan evaluasi agar dapat mengukur kemampuan fitur yang membedakan setiap kelas. Contoh dari penggunaan metode filter yang umum digunakan dalam

proses klasifikasi teks ialah *Document Frequency*, *Mutual Information, Information Gain,* dan C*hi-Square* (Khan et al., 2010).

2.2.4 *Information Gain*

Information Gain merupakan salah satu metode yang digunakan untuk proses seleksi fitur. Nilai suatu *Information Gain* didefinisikan menggunakan *entropy*. Suatu *entropy* menunjukkan berapa banyak informasi yang dibutuhkan saat akan melakukan kode terhadap suatu kelas. *Information Gain* bekerja dengan cara pemberian *scoring* pada tiap nominal atau dengan cara pemberian bobot berupa atribut yang bersifat kontinu kemudian didiskretkan menggunakan nilai maksimal *entropy*. *Information Gain* pada suatu *term* didapat dengan cara menghitung jumlah bit informasi melalui prediksi kategori muncul atau tidaknya suatu term pada suatu dokumen (Maulida et al., 2016).

Information Gain ialah algoritma yang digunakan untuk menentukan batas yang digunakan pada atribut yang tersedia. *Information Gain* melambangkan kualitas atribut yang akan digunakan (Prakoso et al., 2019). Secara umum *Information Gain* terdiri dari tiga tahapan yakni menghitung nilai *Information Gain* setiap atribut yang ada di dataset, menghilangkan atribut yang tidak memenuhi ambang batas, dan memperbaiki atribut dataset yang memenuhi nilai ambang (Bramer, 2007).

Tahapan dalam menghitung bobot nilai IG diantaranya :

1. Hitung bobot nilai entropy dataset melalui persamaan II-20.

$$Entropy(D) = -\sum_{i=1}^{m} P_i \log 2\ (P_i) \qquad \text{(II-20)}$$

Keterangan:

- m : jumlah partisi D
- D : himpunan kasus
- pi : proposi dari Di terhadap D

2. Hitung entropy untuk tiap atribut yang didefinisikan dengan A melalui persamaan II-21.

$$Entropy_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} . Entropy\ (D_j) \qquad \text{(II-21)}$$

Keterangan:

- A : fitur
- D : himpunan kasus
- $|D_j|$: jumlah sampel data pada partisi ke j
- $|D|$: jumlah sampel data
- v : jumlah partisi fitur A
- Entropy(D_j) : total entropy dalam partisi

3. Kemudian langkah terakhir untuk menghitung nilai IG atribut A ialah menggunakan persamaan II-22.

$$Gain(A) = Entropy(D) - |Entropy_A(S)| \qquad \text{(II-22)}$$

Keterangan :

- $Gain(A)$: nilai informasi fitur A
- $Entropy_A(D)$: total *entropy*
- $Entropy_A(S)$: *entropy* A

Setelah mendapatkan bobot nilai akhir *Information Gain* pada tiap atribut maka langkah terakhir ialah dengan memfilter dan mengambil nilai bobot suatu atribut yang memenuhi ambang batas.

2.2.5 *Chi-Square*

Chi-Square merupakan suatu metode seleksi fitur yang digunakan untuk menghilangkan atribut yang kurang relevan pada proses klasifikasi. Seleksi fitur *Chi-Square* mengimplementasikan teori statiska dalam menguji tingkat ipendensi suatu kata berdasarkan kategori dari kata tersebut (Amrullah et al., 2020). Perhitungan *Chi-Square* sendiri terdiri dari beberapa tahapan. Tahapan pertama ialah dengan membuat suatu table kontingensi yang menunjukan relevansi antara suatu *term* dengan kelas yang ada seperti berikut ini.

Tabel II-1. Tabel Relevansi Antara Suatu Kata dan Kelas Kata

Kelas	**Term**	
	t	Not t
c	A	D
Not t	B	C

Kemudian menghitung fungsi *Chi-Square* menggunakan persamaan II-23.

$$x^2(t,c) = \frac{N(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)} \quad \text{(II-23)}$$

Keterangan :

x^2 : Score *Chi-Square*

t : *Term*

c : Kelas

N : Jumlah dokumen latih

A : Total dokumen pada kategori c yang terdapat *term* t

B : Total dokumen pada kategori selain c yang terdapat *term* t

C : Total dokumen pada kategori c yang tidak terdapat *term* t

D : Total dokumen pada kategori selain c yang tidak terdapat *term* t

Setelah mendapatkan nilai *Chi-Square* suatu term pada tiap kategori kelas kata maka diperlukan nilai *Chi-Square* tunggal sebagai skor akhir, dengan memilih satu nilai tunggal yang terbesar diantara nilai *Chi-Square* dikelas kata lainnya. Kemudian skor akhir tersebut akan diurutkan dari yang tertinggi untuk dilakukan proses seleksi fitur (Bahassine et al., 2020).

2.2.6 *Mutual Information*

Mutual Information merupakan salah satu metode yang digunakan untuk melakukan seleksi fitur. MI bekerja dengan cara menghitung banyaknya informasi yang ada pada suatu term, dan kontribusi yang diberikan oleh suatu term dalam

menentukan hasil keputusan pada proses klasifikasi suatu term pada kelas kata secara benar (Hanafi et al., 2020).

Untuk menghitung nilai dari MI pada *term* untuk tiap kategori kelas kata maka dapat menggunakan persamaan II-24.

$$I\ (U,C) = \sum_{et\in\{1,0\}\ ec\in\{1,0\}} P(U = et,\ C = ec)\ log_2 \frac{P\ (U=et,C=ec)}{P\ (U=et)\ P(C=ec)} \qquad \text{(II-24)}$$

Persamaan II-24 dapat dijabarkan kembali menjadi persamaan II-25.

$$I(U,C) = \frac{N_{11}}{N} log_2 \frac{N.N_{11}}{N_1.N_1} + \frac{N_{01}}{N} log_2 \frac{N.N_{01}}{N_0.N_1} + \frac{N_{10}}{N} log_2 \frac{N.N_{10}}{N_1.N_0} + \frac{N_{00}}{N} log_2 \frac{N.N_{00}}{N_0.N_0} \quad \text{(II-25)}$$

Keterangan :

N : Jumlah dokumen yang memiliki et dan ec atau ($N = N_{00} + N_{01} + N_{10} + N_{11}$).

$N_{1.}$: Jumlah dokumen yang memiliki et atau ($N_{1.} = N_{10} + N_{11}$).

N_1 : Jumlah dokumen yang memiliki ec atau ($N_1 = N_{01} + N_{11}$).

$N_{0.}$: Jumlah dokumen yang tidak memiliki et atau ($N_{0.} = N_{01} + N_{00}$).

N_0 : Jumlah dokumen yang tidak memiliki ec atau ($N_0 = N_{10} + N_{00}$).

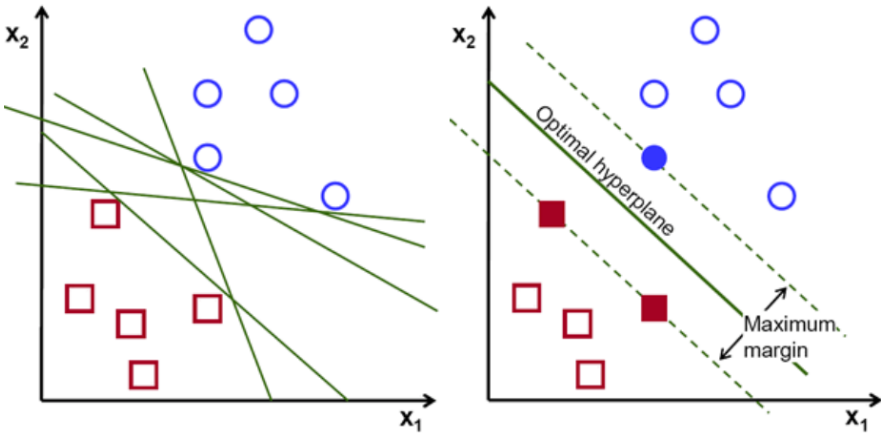
Setelah dilakukan proses perhitungan nilai MI pada suatu kelas, maka selanjutnya ialah menghitung nilai MI pada kelas lain dengan cara yang sama. Kemudian hasil nilai MI pada tiap kelas dibandingkan, nilai MI yang terbesar akan disimpan sebagai hasil akhir. Nilai akhir MI yang didapat pada masing-masing fitur akan diurutkan dari yang terbesar hingga terkecil (Irham et al., 2019).

2.2.7 *Support Vector Machine* (SVM)

Support Vector Machine (SVM) merupakan suatu algortima *supervised learning* yang cukup populer digunakan untuk pengklasifikasian. Algoritma SVM bekerja dengan cara mencari *hyperlane* dengan *margin* yang terbesar. Margin merupakan jarak diantara *support vector* dan *hyperlane*. Dengan memaksimalkan nilai margin, dapat membuat *hyperlane* menjadi lebih baik (Yulietha et al., 2017). Sebagai algoritma klasifikator, SVM termasuk algoritma yang tergolong biner, linier dan probabilistik. SVM menentukan hasil klasifikasi dari data *training* menggunakan *decision boundary* atau batas keputusan. *Decision boundary* bertujuan untuk mencari sebuah model linear atau *hyperlane* teroptimal dalam proses klasifikasi (Mutawalli et al., 2019).

SVM membagi data yang telah diketahui sebelumnya berdasarkan klasifikasi untuk menguji tingkat keakuratan data di sebuah sistem. Dalam penerapannya algoritma SVM digunakan untuk membagi data yang bersifat *non-linear.* Kemudian SVM akan membaginya kedalam *hyperline* yang memisahkan titik *vector.* (Tuhenay, 2021).

Secara umum cara kerja dari SVM dapat tergambarkan pada Gambar II-1 dibawah ini.



Gambar II-1. Ilustrasi Pola SVM

Untuk menghitung persamaan garis *hyperlane* dapat menggunakan persamaan II-3.

$$wi.\ x + b = 0 \qquad \text{(II-3)}$$

Apabila data termasuk kelas positif maka dapat digunakan rumus persamaan II-4.

$$wi.\ x + b \geq +1 \qquad \text{(II-4)}$$

Untuk data yang termasuk kelas negative dapat menggunakan rumus persamaan II-5.

$$wi.\ x + b \leq -1 \qquad \text{(II-5)}$$

Dengan adanya dua garis pemisah tersebut maka dapat menghasilkan persamaan II-6.

$$y_i(\overrightarrow{x_i}.\vec{w} + b - 1) \geq 0 \qquad \text{(II-6)}$$

Margin yang terletak diantara *hyperlane* dan garis pemisah dapat dirumuskan dengan $\frac{1}{||w||} + \frac{1}{||w||} = \frac{2}{||w||}$. Nilai margin dapat dimaksimalkan dengan cara meminimalkan nilai dari ||w|| sebagai penyebut. *Quadratic Programming*

merupakan suatu cara untuk mencari titik minimal dari ||w||, dengan persamaan II-7.

$$min\frac{1}{2}||w||^2 \quad \text{(II-7)}$$

Salah satu teknik permasalahan *Quadratic Programming* yang dapat digunakan adalah *Lagrange Multiplier*. *Lagrange Multiplier* dapat didefinisikan dengan persamaan II-8.

$$L(\vec{w}, b, a) = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} a_i(y_i((\overrightarrow{x_i}.\vec{w} + b) - 1) \quad \text{(II-8)}$$

$$i = 1,2, \dots, n$$

Lagrange Multipliers memiliki koefisien dengan nilai nol atau positif ($a_i \geq 0$). Nilai maksimal dari *Lagrange Multipliers* dapat diketahui dengan meminimalkan nilai variable $\overrightarrow{w}$ dan b kemudian memaksimalkan nilai variable a_i melaui persamaan II-9 dan II-10.

$$\text{L(a)} = \sum_{i=1}^{l} a_i - \frac{1}{2}\sum_{ij=1}^{n} a_i\, a_j y_i y_j \overrightarrow{x_i}\,\overrightarrow{x_j} \quad \text{(II-9)}$$

Dengan syarat :

$$a_i \geq 0, (i = 1,2, \dots, n) \sum_{i=1}^{n} a_i y_i = 0$$

Keterangan :

n : Jumlah data

x : Data

y : Kelas data

Penggunaan persamaan diatas hanya dapat digunakan ketika diperkirakan *hyperlane* dapat memisahkan dua kelas yang terpisah secara *linear* dengan

sempurna. Akan tetapi, nyatanya tidak semua kelas dapat dipisahkan oleh *hyperlane* secara linear (*non linear separable)*. Sehingga proses optimasi pada persamaan (II-6) tidak dapat dilakukan karna batasan tidak berlaku.

Permasalahan tersebut dapat diatasi dengan teknik *softmargin*. Teknik *softmargin* dapat bekerja dengan cara memodifikasi persamaan yang ada pada persamaan (II-6) dan (II-7) dengan menambah variable *slack* ξi ($\xi i > 0$) sebagai berikut :

$$y_i(\overrightarrow{x_i}.\vec{w} + b - 1) \geq 1 - \xi_i, \forall_i \quad \text{(II-10)}$$

$$\frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \xi_i \quad \text{(II-11)}$$

Nilai C digunakan untuk mengontrol penggunaan *trade off* antara *error* klasifikasi ξ dan *margin.* Penggunaan variable C yang semakin besar akan memberikan penalti untuk kesalahan menjadi semakin besar. Dengan menambahkan variable C dan *slack* ξ_i akan memodifikasi persamaan (II-9) menjadi bentuk persamaan II-12.

$$0 \leq \alpha_i \leq C, (i = 1,2,..,n) \quad \text{(II-12)}$$

Proses algoritma pelatihan dapat dilakukan dengan persamaan II-13 sebagai berikut.

$$L(a) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} a_i a_j \, y_i y_j \mathrm{K}(x_i, x_j) \quad \text{(II-13)}$$

Dengan syarat.

$$0 \leq a_i \leq C, \ (i = 1,2,..,n), \ \sum_{i=1}^{n} \alpha_i y_i = 0 \quad \text{(II-14)}$$

Fungsi f(x) sebagai fungsi pemisah optimal didefinisikan melalui persamaan II-15 dan II-16.

$$f(x) = sgn(\sum_{i=1}^{nsv} \alpha_i \, y_i K(x_i, x_d) + b) \quad \text{(II-15)}$$

$$b = \frac{1}{NSV} \sum_{x_j \in SV} \left(\frac{1}{y_i} - \sum_{x_j \in SV} (\alpha_j y_j K(x_j, x_i)) \right) \quad \text{(II-16)}$$

Keterangan :

a : Nilai *Lagrange Multiplier* data *support vector*

nsv : Jumlah *support vector*

x : Data *support vector*

b : Bias

y : Kelas data

x_d : Data uji

$K(x_i, x_j)$: Fungsi kernel

sgn () : Fungsi untuk menentukan tanda bilangan riil

Fungsi *kernel* SVM yang sering digunakan diantaranya :

- Linear

$$K(x_i, x) = {x_i}^T . x \quad \text{(II-17)}$$

- Polinomial

$$K(x_i, x) = (\gamma({x_i}^T . x) + 1)^d \quad \text{(II-18)}$$

- *Radial Basis Function* (RBF)

$$K(x_i, x) = \exp(-y||x_i - x||^2) \quad \text{(II-19)}$$

Keterangan :

γ : Nilai gamma

d : Derajat Polinom

2.2.8 *Multiclass* SVM

Support Vector Machine diketahui hanya dapat melakukan klasifikasi data kedalam dua kelas. Untuk dapat melakukan klasifikasi menjadi lebih dari dua kelas maka dapat dilakukan dengan dua metode pendekatan yaitu *one againts one* vs *one againts all*. Metode *one against one* terdiri dari $\left(\frac{n(n-1)}{2}\right)$ buah model SVM. Sedangkan metode *one againts all* terdiri dari *n* buah model, yang mana *n* merupakan banyaknya kelas (Saniyah, 2019).

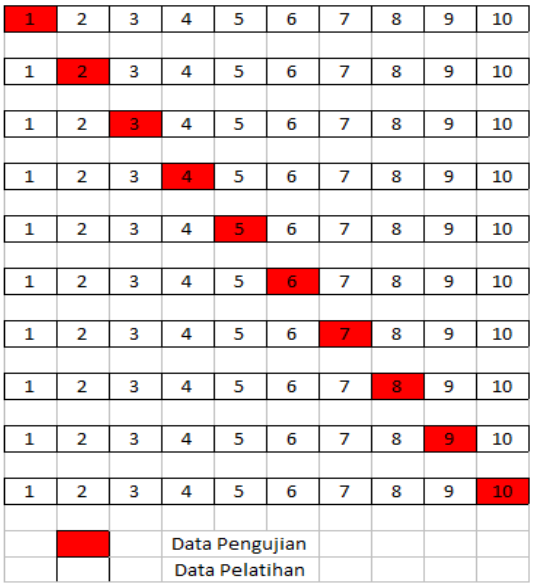
Penelitian ini akan menggunakan metode *one against one* untuk mengklasifikasi kelas pada dokumen pertanyaan berbahasa Indonesia. Setiap model klasifikasi didapatkan melalui proses pelatihan terhadap dua kelas. Kemudian, data uji dimasukkan ke dalam fungsi keputusan $f(x)$. Setelah model klasifikasi telah selesai dibangun maka dapat dilakukan metode voting (Alita et al., 2020). Ketika hasil voting menunjukkan data suatu dokumen termasuk kedalam kelas *i* maka jumlah *vote* terhadap kelas i ditambahkan satu. Hasil klasifikasi ditentukan berdasarkan jumlah hasil voting terbanyak terhadap semua model yang telah dibangun.

2.2.9 *K-Fold Cross Validation*

Salah satu metode yang digunakan untuk mengecek terjadinya *overfitting* pada model yang digunakan ialah dengan *k-fold cross validation*. *Overfitting* sendiri terjadi apabila terdapat penyimpangan yang cukup jauh pada prediksi suatu data. Data yang dibagi menjadi *k* bagian mengizinkan setiap bagian data berhenti untuk

memprediksi data lebih cepat dibandingkan jika tidak dibagi terlebih dahulu (Nasution & Hayaty, 2019).

Pada penggunaan metode *k-fold cross validation*, kumpulan data dibagi sebanyak *k* buah bagian secara acak. Kemudian, dilakukan sebanyak *k*-kali percobaan dengan menggunakan data bagian ke-*k* sebagai data pengujian dan menggunakan sisa bagian lain sebagai data pelatihan. Percobaan yang dilakukan menyesuaikan dengan jumlah bagian yang dilakukan (Supartini et al., 2017)



Gambar II-2. Ilustrasi *k-fold cross validation*

2.2.10 *Confusion Matrix*

Dalam mengukur hasil dan mengevaluasi hasil kinerja metode klasifikasi ada suatu metode yang dikenal dengan *Confusion Matrix* atau sering juga disebut table *confusion*. *Confution Matrix* dapat merepresentasikan kelas aktual dan hasil prediksi dari model klasifikasi. Pada penelitian ini menggunakan *Confusion Matrix*

yang terdiri dari tiga kelas yaitu kelas *factoid*, *non-factoid* dan *others*. Tiga kelas yang terdapat pada *Confusion Matrix* dapat membentuk *matrix* dengan ukuran 3x3 dan menghasilkan sembilan sel *matrix* (Irmanda & Astriratma, 2020). Model *Confusion Matrix* 3x3 dapat di ilustrasikan seperti pada tabel II-2.

Tabel II-2. Model *Confusion Matrix*

Fakta	**Prediksi**		
	A	**B**	**C**
A	TA	FB1	FC1
B	FA1	TB	FC2
C	FA2	FB2	TC

Keterangan *confusion matrix* :

TA : Kelas A yang diklasifikasikan dengan benar

TB : Kelas B yang diklasifikasikan dengan benar

TC : Kelas C yang diklasifikasikan dengan benar

FA1 : Kelas B yang diklasifikasikan kedalam kelas A

FA2 : Kelas C yang diklasifikasikan kedalam kelas A

FB1 : Kelas A yang diklasifikasikan kedalam kelas B

FB2 : Kelas C yang diklasifikasikan kedalam kelas B

FC1 : Kelas A yang diklasifikasikan kedalam kelas C

FC2 : Kelas B yang diklasifikasikan kedalam kelas C

Setelah matrix 3x3 dibentuk, ada beberapa kriteria kinerja yang dapat diukur dalam *Confusion matrix* yaitu *Accuracy, Recall, Precision*, dan *F-measure*. Pada kasus klasifikasi *multiclass* nilai *average score* yang diambil dapat berupa *average score* ("*macro", "micro"* dan "*weighted"*). *Average score* yang digunakan pada penelitian ini ialah *macro*. Berikut beberapa persamaan untuk kriteria kinerja tersebut.

a. *Accuracy* bertujuan untuk menghitung tingkat ketepatan suatu model dalam melakukan klasifikasi data. Perhitungan *accuracy* ditunjukkan pada persamaan II-26.

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad \text{(II-26)}$$

b. *Recall* bertujuan untuk menghitung kemampuan model dalam menemukan informasi yang berhubungan. Perhitungan nilai *recall* ditunjukkan pada persamaan II-27.

$$Recall = \frac{TP}{TP + FN} \quad \text{(II-27)}$$

c. *Precision* merupakan perbandingan antara total dokumen yang saling berhubungan dengan total dokumen keseluruhan yang digunakan dalam model klasifikasi. Perhitungan nilai *precision* ditunjukkan pada persamaan II-28.

$$Precision = \frac{TP}{TP+FP} \quad \text{(II-28)}$$

d. *F-Measure* bertujuan untuk menunjukkan keseimbangan yang terjadi antara nilai *recall* dan nilai *precision*. Perhitungan nilai *F-measure* ditunjukkan pada persamaan II-29.

$$F - measure = \frac{2\ X\ (Precision\ X\ Recall)}{Precision + Recall} \quad \text{(II-29)}$$

2.2.11 *Rational Unified Process*

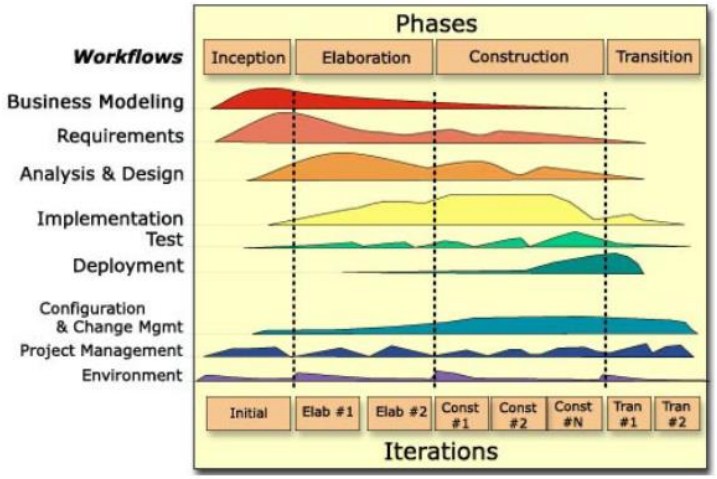
Rational Unified Process (RUP) merupakan suatu proses yang bersifat iteratif dalam pengembangan rekayasa perangkat lunak. Proses ini dilakukan melalui pendekatan secara disiplin dalam menetapkan suatu tugas dan tanggung jawab dalam suatu organisasi pengembang perangkat lunak. Tujuan dari proses ini ialah untuk memastikan bahwasanya produksi dari perangkat luank yang telah dikembangakn memiliki kulaitas yang tinggi dan sesuai dengan harapan dari *end-user*. Karakteristik dari metode RUP ialah menggunakan menggunakan metodology OOP (*Object Oriented Programming*), menggunakan pendekatan bersifat iteratif untuk *lifecycle* dalam pengembangan perangkat lunak, serta model pengembangan menggunakan UML (*Unified Model Language*). RUP dalam satu siklus, RUP terdiri dari 4 fase diantaranya *Inception*, *Elaboration*, *Construction*, dan *Transition* (Pessoa & Anwar, 2014).

1. *Inception* merupakan fase awal pada proses RUP yang bertujuan untuk mendapatkan dan menganalisa kebutuhan dari semua pihak yang terkait. Pada fase ini dilakukan beberapa hal seperti membuat suatu *business case*, dan menetapkan ruang lingkup dari system yang akan dikembangkan.
2. *Elaboration* merupakan fase yang dikerjakan setelah melalui fase *inception.* Pada tahap ini dilakukan analisa manajemen resiko, menetapkan *base line* atau aturan kesepakatan, dan dilakukan analisa kebutuhan sistem.
3. *Construction* merupakan fase dimana proses pengembangan sistem yang telah dianalisa sebelumnya, di implemtasi di fase ini. Proses implementasi

yang dilakukan melibatkan proses diantaranya analisa, desain, *prototype* dan *testing*.

4. *Transition* merupakan tahapan akhir dari fase yang ada di RUP. Fase ini berfokus pada peluncuran (*launching)* dari suatu produk, pembuatan dokumentasi dari produk yang dikembangkan dan juga pemeliharaan (*maintenance*) dari perangkat lunak.

Rangkaian fase RUP dapat dilihat pada gambar II-3.



Gambar II-3. Fase *Rational Unified Process* (RUP)

2.3 Penelitian Lain yang Relevan

Pada penelitian ini terdapat uraian mengenai beberapa penelitian relevan yang bersumber dari prosiding dan jurnal ilmiah dari peneliti lain. Dalam rangka menambah referensi bagi peneliti dalam memperkuat landasan berpikir dalam menyelesaikan masalah pada topik yang diangkat.

Pada penelitian yang dilakukan oleh (Lucia & Londo, 2019), mempelajari beberapa algoritma klasifikasi yang digunakan untuk mengklasifikasi artikel

berbahasa Indonesia. Ada banyak model *Machine Learning* yang dapat digunakan untuk melakukan tugas tersebut. Untuk itu, peneliti membandingkan penggunaan beberapa model *Machine Learning* yang cukup popular diantaranya *Multinomial Naïve Bayes, Decision Tree* dan *Support Vector Machine*. Hasil penelitian menunujukkan SVM mendapatkan total akurasi terbaik sebesar 93%. Kemudian, diikuti MNB dengan akurasi berkisar 85-88%. *Decision Tree* memperoleh hasil terkecil karna hanya mendapatkan hasil akurasi sebesar 80-81%.

(Made et al., 2018) melakukan pengklasifikasian untuk mengidentifikasi teks tang mengandung konten *bullying*. SVM digunakan sebagai metode klasifikasi untuk mencari *hyperlane* pada kelas positif dan negative. Peneliti juga menggunakan metode seleksi fitur *Information Gain* untuk menyeleksi fitur dengan tingkat relevansi yang rendah. Hasil dari penelitian menggunakan metode SVM dengan nilai percobaan *iterMax* = 20, λ = 0.5, γ = 0.001, ε = 0.000001, dan C = 1. Untuk penggunaan *threshold* terbaik pada seleksi fitur *Information Gain* ialah 90%. *Threshold* ini mendapatkan hasil kinerja berupa 76.66% *accuracy*, 72,22% *precision*, *recall* 86.66% dan *f-measure* 78.78%.

(Luthfiana et al., 2020) dalam penelitiannya, melakukan klasifikasi untuk analisis sentiment berupa *user feedback* pada aplikasi. Penlitian ini membagi dataset menjadi 3 kelas yaitu positif, negatif dan netral. Berdasarkan hasil uji coba yang telah dilakukan dengan rasio pembagian data 80:20 dan penggunaan *threshold Chis-Square* sebesar 6,63 mendapatkan hasil kinerja terbaik berupa *accuracy 77%*, *precision* 50%, *recall* dan *F1-Score* 73%. Untuk pengklasifikasian dengan SVM dengan parameter terbaik yaitu C 100 dan gamma 0,001. Untuk proses klasifikasi

tanpa menggunakan *Chi-Square* hanya mendapatkan hasil kinerja terbaik dengan parameter C 10 dan gamma 0,01 berupa *accuracy* sebesar 69%, *precision* 48%, *recall* 53% dan *F-1 Score* 50%.

(Hanafi et al., 2020) meneliti klasifikaisi multilabel pada dokumen hadis bukhari dalam terjemahan Bahasa Indonesia. Proses klasifikasi menggunakan model unigram/bigram, seleksi fitur menggunakan *Mutual Information* (MI) dan algoritma klasifikasi menggunakan *Support Vector Machine* (SVM). Pada penelitian ini metode seleksi fitur *Mutual Information* mampu meningkatkan *accuracy* dari 91,86% menjadi 93,4%.

Perbandingan metode seleksi fitur dan algoritma dalam rangka optimasi kinerja teks sebelumnya telah dilakukan oleh (Chandra, 2019) dengan membandingkan beberapa metode seleksi yang cukup terkenal diantaranya *Chi-Square, Term Frequency* dan *Mutual Information*. Selain itu peneliti juga melakukan perbandingan terhadap algortima klasifikasi yaitu SVM dan MNB. Dataset yang digunakan ialah dokumen tidak seimbang pada text Indonesia. Hasil pengujian mendapatkan skor akhir mulai dari 85 hingga 90 di tiap F1-Score. Metode ini diuji pada 2 data set yaitu pada *standard benchmarking dataset* dan dataset berupa teks berita Indonesia. Pada pengujian ini, metode seleksi fitur *Chi-Square* mendapatkan hasil yang paling konsisten dengan hasil rata-rata yaitu 89.76% saat dikombinasikan menggunakan algoritma SVM.

(Sahuri, 2018) melakukan studi perbandingan untuk memperoleh fitur terbaik dan metode seleksi yang cocok dengan metode klasifikasi tertentu. Metode

seleksi fitur yang dibandingkan dalam penelitian ini ialah *Information Gain*, *Gini Index*, dan *Chi-Square*. Selanjutnya, untuk metode pengklasifikasian peneliti membandingkan metode klasifikasi *Neural Network*, *K-Nearest Neighbor*, *Naïve Bayes* dan *Support Vector Machine*. Hasil menunjukkan metode klasifikasi *K-Nearest Neighbor* memperoleh hasil maksimal ketika dikombinasikan dengan metode seleksi fitur *Information Gain* dengan nilai k sebesar 4. *Neural Network* juga menghasilkan nilai optimal jika dikombinasikan dengan *Information Gain.* Sedangkan klasifikasi *Support Vector Machine* memperoleh hasil maksimal yakni sebesar 82% ketika diuji menggunakan seleksi fitur *Chi-Square* dan *Information Gain*.

2.4 Kesimpulan

Bab ini telah menguraikan landasan teori yang digunakan dan juga menguraikan hasil penelitian-penelitian sebelumnya yang berhubungan dengan konsep penelitian yang dilakukan yaitu klasifikasi pertanyaan berbahasa indonesia menggunakan seleksi fitur IG, *Chi-Square* dan MI. Kemudian, menggunakan SVM sebagai algortima klasifkasi. Landasan teori yang dicantumkan dalam bab ini akan menjadi referensi pada bab selanjutnya.