BAB III
METODELOGI PENELITIAN

3.1 Subjek dan Objek Penelitian

Subjek penelitian merupakan pengguna atau orang yang memberikan informasi terkait data yang dibutuhkan. Subjek penelitian ini adalah *review* pengguna dari *ecommerce* Tokopedia dan Lazada. Sedangkan objek penelitian adalah penanganan *slang*word dalam analisis sentimen.

3.2 Alat dan Bahan

3.2.1 Perangkat Keras

Dalam penelitian ini, digunakan sebuah laptop dengan konfigurasi berikut:

1. CPU Intel(R) Core (TM) i3
2. Memori RAM sebesar 8 GB
3. Kapasitas penyimpanan SSD 256GB
4. Layar berukuran 14” Full HD

3.2.2 Perangkat Lunak

Beberapa perangkat lunak yang digunakan dalam penelitian ini termasuk:

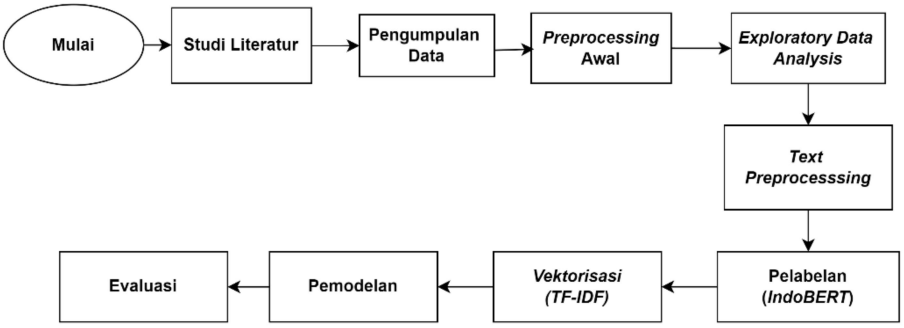
1. Sistem operasi *Windows*
2. *Google Colaboratory*
3. Peramban web

3.2.3 Bahan

Penelitian ini menggunakan bahan berupa kumpulan data yang diperoleh dari platform penyedia data yaitu Kaggle. Dataset yang digunakan adalah data product *review* dari Lazada dan Tokopedia.

3.3 Alur Penelitian

Beberapa tahapan yang dilakukan untuk menyelesaikan penelitian penerapan kamus *slang word* dalam klasifikasi teks bahasa indonesia diilustrasikan pada diagram alir seperti Gambar 3.1.



Gambar 3. 1 Diagram Alur Penelitian

3.3.1 Studi Literatur

Studi literatur merupakan tahapan awal penting dalam proses penelitian, di mana peneliti mencari, menganalisis, dan menyintesis sumber-sumber literatur terkait topik penelitian sebagai acuan dalam membangun konsep dan metode dalam penelitian ini. analisis literatur yang dilakukan mencakup pemahaman dan tinjauan literatur yang mendalam terhadap publikasi penelitian terkait normalisasi teks dalam klasifikasi sentimen. Dalam tahap studi literatur, peneliti mengkaji publikasi penelitian tentang normalisasi text dengan kamus *slang word*, mengkaji pengaruhnya terhadap akurasi model klasifikasi, mengkaji kelemahan dan kelebihan penerapan korpus *slang word*.

3.3.2 Pengumpulan Data

Dalam penelitian ini, sumber data yang digunakan merupakan data sekunder, yang berasal dari penelitian atau sumber lainnya. Data yang digunakan diambil dari platform *Kaggle*, yang menyediakan data untuk keperluan penelitian dan analisis. Dataset yang dianalisis dalam penelitian ini terdiri dari ulasan produk di platform Lazada [30] dan data *Product Reviews Category Food and Drink* dari platform Tokopedia [31]. Data Lazada yang diperoleh dari Kaggle berjumlah sekitar 203.787. Sedangkan pada Tokopedia yang diperoleh dari Kaggle berjumlah sekitar 5934. Sampel data tokopedia dan lazada dapat dilihat pada Tabel 3.1 dan Tabel 3.2.

Tabel 3. 1 Review Tokopedia

No	*Review*
1.	Barangnya nyampe super cepet nih, mantul banget, makasih Lazada!
2.	Pesenan sampe cepet banget, packingnya rapi abis, harganya murah lagi, *plus free* ongkir!
3.	Keren parah barangnya, sesuai pesenanku banget!
4.	Jangan deh beli disini, parah banget, udah dua minggu lebih barangnya gak sampe, alasan-alasannya ga jelas, gangguan mulu lah!
5.	Hati-hati ya, TV ini gak bisa nyetel pake USB loh!

Tabel 3. 2 Review Lazada

No	*Review*
1.	Best. Paling enak dibanding produk sejenis. Ngga eneg
2.	Dulu sempet nemenin dia ke tempat kerja beda, beliin dia salmon mentai di Momomaru kalo ga salah. Tahun lalu dia pengen Momomaru lagi, baru bisa beliin sekarang, pas dia udah satu tim sama gue. Meskipun ongkirnya lebih gedean dari harga satu porsi salmon mentai, gue tetep mau bayar utang gue, walaupun sekarang dia udah deket sama orang lain.
3.	Kimcinya enak banget rasanya, gak terlalu nendang.
4.	Seru abis belanja di sini, layanannya keren banget!
5.	Udah beberapa kali pesen, rasanya selalu oke dan sesuai harganya.

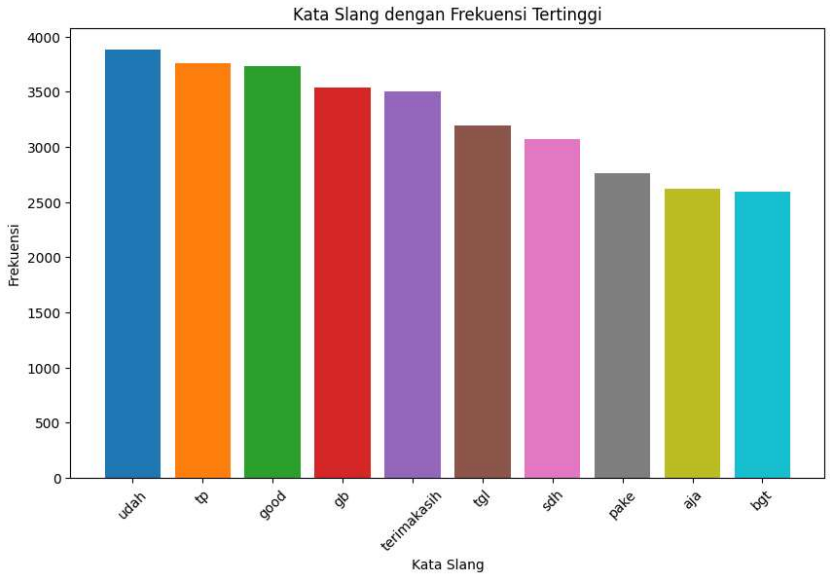
Dalam penelitian ini terdapat beberapa sumber kamus *slang word* yang tersedia di internet. Penelitian akan menggunakan kamus *slang word* yang bersumber dari platform *Github* [32][33]. Kamus *slang word* [32] diperoleh berdasarkan data dengan topik “anisbaswedan” dan “ahok” pada twitter, diperoleh sebanyak 1294 kata *slang*. Sedangkan sumber [33] menghasilkan sebanyak 15803 kata *slang.* Selain itu, terdapat kumpulan kata bahasa Indonesia yang diperoleh dari platform *Github* [34] dengan total sebanyak 110.407 data.

3.3.3 *Preprocessing* Awal

Tahap *preprocessing* Awal dalam penelitian ini dilakukan dengan penghapusan *missing value* dan penghapusan data *duplicate.* Data Lazada yang diperoleh dari *Kaggle* awalnya berjumlah 203787 ribu. Namun, dalam penelitian ini data diolah terlebih dahulu dengan menghapus *missing value* dan menghapus data duplikat sebelum dianalisis. Sehingga, data lazada memiliki 38066 ribu *review*. Penghapusan *missing value* dan duplikat juga dilakukan pada data Tokopedia yang pada awalnya *value* data berjumlah 5934 setelah dilakukan penghapusan data dan duplikat *review* menjadi 4974 *review*.

3.3.7 *Exploratory Data Analysis*

Dalam tahap ini juga dilakukan visualisasi frekuensi kata pada dataset. Ilustrasi visualisasi dapat dilihat pada Gambar 3.2



Gambar 3. 2 Visualisasi Bar Chart Frekuensi Kata *Slang*

Gambar 3.2 memperlihatkan frekuensi beberapa kata slang yang terdapat pada dataset, seperti "sdh", "gak", "tp", dan "cepet". Kata-kata ini merupakan kata *slang* atau *noise* yang sering muncul dalam dataset. Deteksi frekuensi yang tinggi dari kata-kata slang ini menunjukkan bahwa masih banyak kata *slang* yang

terdeteksi, sehingga perlunya dilakukan normalisasi teks untuk meningkatkan akurasi analisis sentimen.

3.3.6 Text Preprocessing

Tahapan *Preprocessing* data merupakan tahapan persiapan dimana data yang akan digunakan yang berupa teks diolah untuk analisis sentimen. Dataset yang tersedia masih berupa data mentah, sehingga perlu dilakukan proses *cleaning* data untuk porses selanjutnya. Data *Preprocessing* memiliki fungsi utama Untuk memverifikasi keakuratan dan konsistensi data awal yang akan diproses, sehingga hasil analisis dapat dianggap valid. [35] . Dalam penelitian ini, dua skenario *preprocessing* akan diterapkan. Pada skenario pertama, dilakukan *preprocessing* standar yang mencakup langkah-langkah dasar seperti penghapusan karakter khusus, *stop words*, dan tokenisasi teks. Skenario ini bertujuan untuk mempersiapkan data dalam bentuk yang lebih terstruktur dan siap untuk dianalisis oleh model *machine learning*. Sementara itu, pada skenario kedua, selain langkah-langkah standar, dilakukan juga normalisasi kata *slang*. Normalisasi ini bertujuan untuk mengganti kata-kata slang dengan padanan kata yang lebih formal dan memperbaiki kesalahan penulisan yang umum terjadi. Implementasi kedua skenario ini memungkinkan untuk mengevaluasi pengaruh normalisasi dan pembersihan tambahan terhadap performa model sentimen analisis.

1. **Skenario *Preprocessing* Standar**

Pada skenario pertama ditetapkan preprocessing standar yaitu dimuali dari *case folding, filtering, tokenizing, stopword removal*.

a. Case Folding

Case folding adalah proses yang mengubah semua karakter huruf dalam dokumen menjadi huruf kecil *(lowercase)*.

Tabel 3. 3 Hasil Case Folding

Data	Hasil *Case Folding*
Baguss bangett kurir nya jga ramah, cakep pula wkwk. Ini 64gb tp cmn 58 sekian gb tp gpp lah yah... Pokok nya recomended deh....	baguss bangett kurir nya jga ramah, cakep pula wkwk. ini 64gb tp cmn 58 sekian gb tp gpp lah yah... pokok nya recomended deh....

Berdasarkan Tabel 3.3 data *review* yang memiliki karakter *uppercase* (huruf besar) diubah menjadi huruf kecil melalui tahap *case folding.* Huruf "B", "I", dan "P" pada *review* di atas diubah menjadi huruf kecil "b", "i", dan "p".

b. Cleaning

Menghapus link, tab, number, punction and single char. Pada tahap ini hasil dari *case folding* akan dilakukan pengahapusan *punction, link, tab, number dan single char.*

Tabel 3. 4 Hasil Cleaning

Hasil *Case Folding*	Hasil *Cleaning*
baguss bangett kurir nya jga ramah, cakep pula wkwk. ini 64gb tp cmn 58 sekian gb tp gpp lah yah... pokok nya recomended deh....	baguss bangett kurir nya jga ramah cakep pula wkwk ini gb tp cmn sekian gb tp gpp lah yah pokok nya recomended deh

Tabel 3.4 menampilkan hasil dari *case folding* akan dilakukan proses *cleaning* dengan menghapus tanda baca*(punction), number, single char* yang nantinya akan disimpan pada variabel hasil *cleaning* pada dataframe.

c. Tokenizing

Tokenizing bertujuan untuk menghilangkan tanda, tag, emoticon, dan memotong setiap kalimat ke dalam sekumpulan kata tunggal.

Tabel 3. 5 Hasil Case Tokenizing

Hasil *Cleaning*	Hasil *Tokenize*
baguss bangett kurirnya jga ramah cakep pula wkwk ini gb tp cmn sekian gb tp gpp lah yah pokok nya recomended deh	[baguss, bangett, kurir, nya, jga, ramah, cakep, pula, wkwk, ini, gb, tp, cmn, sekian, gb, tp, gpp, lah, yah, pokok, nya, recomended, deh]

Data hasil dari *cleaning* akan dilakukan proses tokenisasi untuk memecah kalimat tersebut menjadi beberapa bagian. Hasil proses tokenisasi dapat dilihat pada Table 3.5.

d. Stopword Removal

Stopword Removal merupakan proses untuk menghapus kata – kata yang tidak memiliki makna pada suatu kalimat. Contoh kata – kata yang bisa dihapus

seperti di, ke, itu, yang, dan lain sebagainya. Untuk stopword ini menggunakan *library nltk* berbahasa Indonesia.

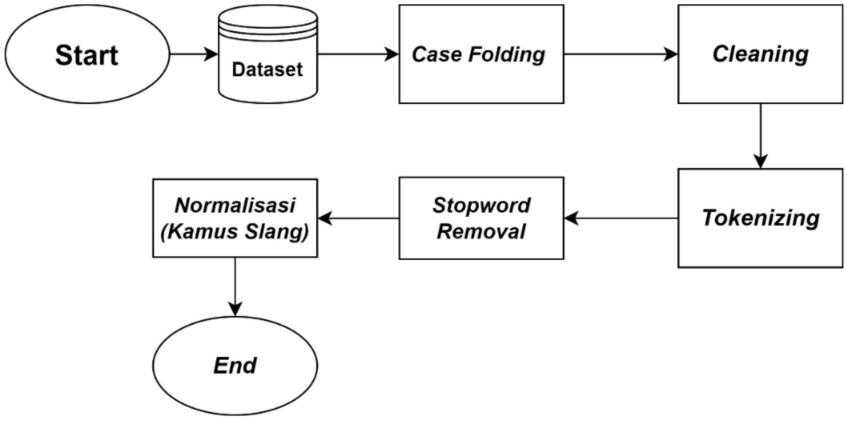
Tabel 3. 6 Hasil Stopword Removal

Hasil *Tokenize*	Hasil *Stopword Removal*
[baguss, bangett, kurir, nya, jga, ramah, cakep, pula, wkwk, ini, gb, tp, cmn, sekian, gb, tp, gpp, lah, yah, pokok, nya, recomended, deh]	[baguss, bangett, kurir, jga, ramah, cakep, pula, wkwk, gb, tp, cmn, sekian, gb, tp, gpp, lah, pokok, recomended]

Proses *stopword removal* atau menghapus kata yang tidak memiliki makna dilakukan menggunakan hasil dari proses tokenisasi. Hasil *stopword removal* dapat dilihat pada Tabel 3.6

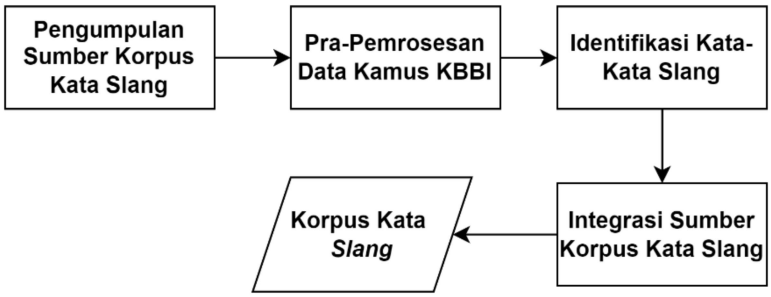
e. Normalisasi

Tahapan Normalisasi dalam analisis sentimen yaitu mengubah kata -kata yang mengandung *slang* menjadi kata formal sesuai dengan KBBI. Tahap normalisasi dilakukan setelah tahapan *stopword removal* dengan menerapkan korpus *slang* word yang sudah dibuat. Langkah – langkah dalam skenario kedua dapat dilihat pada Gambar 3.3



Gambar 3. 3 Tahapan Preprocessing dengan Normalisasi

Dalam skenario normalisasi ini, korpus kata *slang* yang digunakan bersumber dari dua sumber utama yaitu kamus *slang* dari internet dan kata *slang* yang diidentifikasi dari dataset *review* produk Lazada dan Tokopedia. Alur pembuatan korpus kata slang dapat dilihat pada Gambar 3.4.



Gambar 3. 4 Alur Pembuatan Kopus Kata Slang

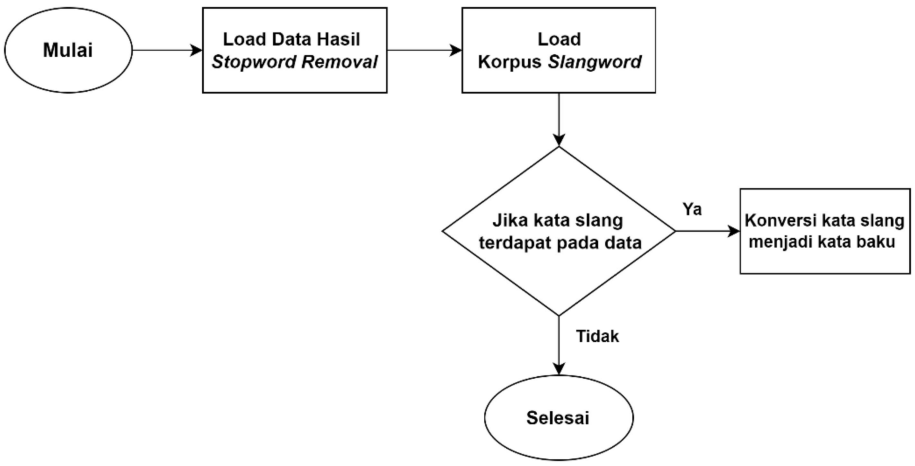
Berdasarkan Gambar 3.4 Proses pengumpulan sumber kamus *slang* dimulai dengan mengumpulkan daftar kata *slang* dari berbagai sumber online [32][33]. Pada tahap pra-pemrosesan, data dari Kamus Besar Bahasa Indonesia (KBBI) diambil dan dipersiapkan sebagai referensi untuk kata-kata formal, memastikan bahwa istilah *slang* dapat diubah ke dalam bentuk resmi yang sesuai. Analisis dataset *review* produk dari Lazada dan Tokopedia dilakukan untuk mengidentifikasi kata-kata *slang* yang ada, termasuk istilah yang tidak standar atau berbentuk singkatan. Selanjutnya, integrasi dilakukan dengan menggabungkan daftar kata *slang* dari sumber online serta kata *slang* yang diidentifikasi dari dataset, menghasilkan korpus *slang*. Korpus kata slang yang telah terintegrasi disiapkan untuk digunakan dalam proses normalisasi teks, yang berfungsi mengganti istilah *slang* dengan padanan kata formal sesuai dengan KBBI. Tampilan korpus dapat dilihat pada Tabel. 3.7

Tabel 3. 7 Kamus Slang word

Slang	**Formal**
bgs	bagus
mantaaaaaaapppp	mantap
lelet	lambat
mnipu	tipu
males	malas

Terdapat dua dataset yang digunakan yaitu data hasil dari *stop word removal* dan korpus *slang*. Pada langkah ini, setiap kata akan dicek oleh komputer, apabila kalimat tersebut mengandung kata *slang* yang yang cocok dengan pola pada korpus

slang, maka kata tersebut akan dikonversi menjadi kata baku sesuai dengan corpus *slang.* Jika kata tidak cocok dengan kamus *slang*, biarkan kata tersebut tidak berubah dan dilanjutkan pada tahapan *preprocessing* selanjutnya. Langkah – langkah dalam penormalisasian kata *slang* dapat dilihat pada Gambar 3.



Gambar 3. 5 Tahap Normalisasi

Berdasarkan Gambar 3.5 berikut merupakan contoh pseudocode normalisasi kata *slang*:

Algoritma penerapan kamus *slang word*

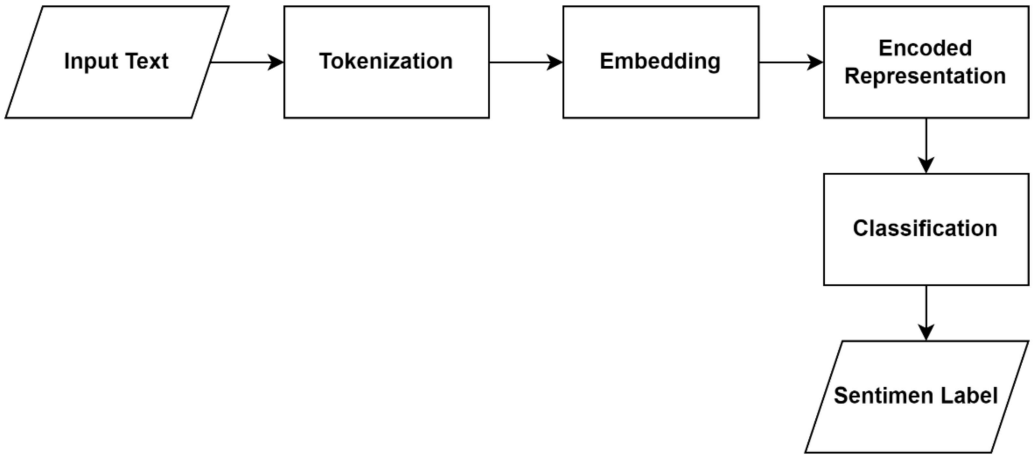
```
def convertTo Slangword(text):
    kamus Slang =eval(open("Slangword.csv").read())
    pattern = re.compile(r'\b( ' + '|'.join(kamus Slang.keys())+r')\b')
    content = []
    for kata in text:
        filter Slang = pattern.sub(lambda x: kamus Slang[x.group()],kata)
        content.append(filter Slang.lower())
    review Content = content
    return review Content

df['Formalisasi'] =
df['review Content_stop_removed'].apply(convertTo Slangword)
```

Kode *pseudocode* di atas bertujuan untuk mengkonversi teks yang mengandung kata *slang* menjadi bentuk formal berdasarkan korpus *kata slang* yang tersimpan dalam file "*Slang*word.csv". kode diatas akan mencocokan kata – kata yang cocok dengan korpus *kata slang word*. Kode di atas akan memproses karakter

per karakter dalam teks. Jika terdapat kecocokan maka akan dikonversi menjadi kata baku sesuai dengan korpus *kata slang*.

3.3.7 Pelabelan *IndoBERT*

Pelabelan sentimen dari data menggunakan model *IndoBERT* memiliki keunggulan dalam menangkap dan menganalisis nuansa bahasa secara lebih mendalam dibandingkan dengan pelabelan manual. Proses pelabelan sentimen dengan *IndoBERT* melibatkan beberapa tahapan penting, mulai dari tokenisasi teks, *embedding*, dan *encoding* hingga klasifikasi akhir untuk menghasilkan label sentimen yang akurat.



Gambar 3. 6 Proses Kerja pelabelan IndoBERT

Berdasarkan gambar 3.2 yang menggambarkan cara kerja *IndoBERT* untuk melabeli sentimen.

1. *Input Text*

Kalimat yang akan dianalisis: "barang bagus banget sesuai gambar dan deskripsi kirim cepat bagus dan rekomendasi banget".

2. *Tokenization*

Kalimat tersebut dipecah menjadi token atau kata-kata, seperti ['barang', 'bagus', 'banget', 'sesuai', 'gambar', 'dan', 'deskripsi', 'kirim', 'cepat', 'bagus', 'dan', 'rekomendasi', 'banget'].

3. *Embedding*

Setiap token diubah menjadi *vektor embedding*, yaitu representasi numerik berdimensi tinggi yang dapat diproses oleh model.

4. *Encoded Representation*

Vektor embedding dari token-token *input* diproses melalui beberapa lapisan *encoder* dari *IndoBERT*.

5. *Classification*

Hasil akhir dari lapisan *encoder* adalah representasi kontekstual dari seluruh kalimat, yang artinya model telah memahami makna dan hubungan antar kata dalam kalimat tersebut. Representasi ini kemudian diteruskan ke lapisan klasifikasi, yang bertugas menentukan label sentimen dari kalimat tersebut.

6. *Sentiment Label*

Lapisan klasifikasi memberikan *output* berupa label sentimen, seperti Positif dan Negatif.

3.3.8 Vektorisasi

Data yang sudah melalui tahap preprocessing selanjutnya dilakukan *feature extraction* untuk mengubah data teks (string) menjadi sebuah vector. Dalam penelitian ini metode *feature extraction* yang digunakan yaitu TF- IDF. Contoh perhitungan TF -IDF dapat dilihat pada Tabel 3.13.

Tabel 3. 8 contoh hasil preprocessing

Dokumen 1	bagus banget kurir juga ramah cantik tawa bosan tapi Cuma sekian tidak apa apa pokok mantap
Dokumen 2	maaf barang tidak terima

Pada Tabel 3.8 terdapat dua dokumen hasil dari tahapan *preprocessing* yang akan digunakan untuk pembobotan term menggunakan TF-IDF. Berikut perhitungan tf-idf dari data hasil *preprocessing*.

1. Menghitung bobot TF setiap kata dalam satu dokumen. *Term frequency* menghitung frekuensi kemunculan term pada setiap dokumen. *Term frequency* dilakukan dengan membandingkan frekuensi dari term tersebut dengan total

frekuensi term pada dokumen. Berikut sampel perhitungan TF pada setiap dokumen yang dapat dilihat pada Tabel 3.9.

Tabel 3. 9 Perhitungan Tf

Term(t)	D1	D2
Bagus	1	0
Banget	1	0
Kurir	1	0
Maaf	0	1
Tidak	1	1

2. Menghitung DF (*Document Frequency).* Perhitungan DF mengacu pada berapa kali suatu kata atau *term* muncul pada seluruh dokumen. Contoh perhitungan DF dapat dilihat pada Tabel 3.10.

Tabel 3. 10 Perhitungan DF

Term(t)	TF		DF
	D1	D2	
Bagus	1	0	1
Banget	1	0	1
Kurir	1	0	1
Maaf	0	1	1
Tidak	1	1	2

Berdasarkan Tabel 3.10 hasil DF dihitung berdasarkan jumlah kemunculan kata dari seluruh dokumen. Term "tidak" memiliki DF sebesar dua dikarenakan term tersebut muncul masing – masing satu pada dokumen satu dan dokumen dua.

3. Tahap selanjutnya yaitu menghitung IDF (*Inverse Document Frequency)* yaitu menghitung *frekuensi* kemunculan kata tersebut di seluruh dokumen, menentukan seberapa tinggi atau rendah nilai IDF-nya; semakin jarang kata tersebut muncul di antara dokumen, semakin tinggi nilai IDF-nya.

Tabel 3. 11 Perhitungan IDF

Term(t)	**TF**		**DF**	**IDF** $\boldsymbol{log = (n/df)}$
	D1	**D2**		
Bagus	1	0	1	$log\frac{2}{1} = 0{,}3010$
Banget	1	0	1	$log\frac{2}{1} = 0{,}3010$

Term(t)	TF		DF	IDF $log = (n/df)$
	D1	D2		
Kurir	1	0	1	$log\frac{2}{1} = 0{,}3010$
Maaf	0	1	1	$log\frac{2}{1} = 0{,}3010$
Tidak	1	1	2	$log\frac{2}{2} = 0$

Berdasarkan Tabel 3.11 term "tidak" memiliki nilai IDF terkecil karena Nilai IDF dihitung sebagai logaritma dari kebalikan dari jumlah dokumen yang mengandung *term* dibagi dengan total jumlah dokumen (N).

4. Menghitung Bobot TF-IDF dengan cara mengkalikan bobot TF dengan IDF pada setiap dokumen.Perhitungan TF-IDF dari setiap kata dapat dilihat pada Tabel 3.12

Tabel 3. 12 Perhitungan TF-IDF

No	Term	TF		DF	IDF $log = (n/df)$	TF-IDF	
		D1	D2			D1	D2
1.	bagus	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
2.	banget	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
3.	kurir	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
4.	Juga	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
5.	ramah	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
6.	cantik	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
7.	tawa	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
8.	bosan	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
9.	tapi	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
10.	cuma	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
11.	sekian	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
12.	tidak	1	1	2	$log\frac{2}{2} = 0$	0	0
13.	apa	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
14.	pokok	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0
15.	mantap	1	0	1	$log\frac{2}{1} = 0{,}3010$	0,3010	0

No	Term	TF		DF	IDF	TF-IDF	
		D1	D2		$log = (n/df)$	D1	D2
16.	maaf	0	1	1	$log\frac{2}{1} = 0{,}3010$	0	0,3010
17.	barang	0	1	1	$log\frac{2}{1} = 0{,}3010$	0	0,3010
18.	terima	0	1	1	$log\frac{2}{1} = 0{,}3010$	0	0,3010

3.3.9 Model Klasifikasi

1. Naive Bayes Classifier

Naïve Bayes adalah sebuah algoritma klasifikasi yang mengandalkan pada probabilitas dan statistik. Pendekatan algoritma ini mengacu pada *teorema Bayes* untuk menggabungkan pengetahuan sebelumnya dengan informasi baru. Algoritma *Naive Bayes*, diasumsikan bahwa fitur-fitur (kata-kata dalam teks) adalah independen satu sama lain. Dalam algoritma *Naive Bayes*, teks harus di *preprocessing* dan direpresentasikan dalam bentuk vektor dengan TF – IDF.

Tabel 3. 13 Label Data latih

Dokumen 1	bagus banget kurir juga ramah cantik tawa bosan tapi cuma sekian tidak apa apa pokok mantap	Positif	15 t*erm*
Dokumen 2	maaf barang tidak terima	Negatif	4 t*erm*

Berdasarkan Tabel 3.14 langkah awal yang dilakukan yaitu:

1. Menghitung probabilitas setiap kategori yaitu kategori positif dan kategori negatif.

$$P(positif) = \frac{f(positif)}{\sum c} = \frac{1}{2} = 0{,}5$$

$$P(negatif) = \frac{f(negatif)}{\sum c} = \frac{1}{2} = 0{,}5$$

2. Menghitung probabilitas setiap term dari semua dokumen. Dari dua dokumen jumlah seluruh kata sebanyak 19 *term* dengan rincian 15 *term* dari kelas positif dan 4 *term* dari kelas negatif.

a. Probabilitas kata “bagus”

$$P(bagus|positif) = \frac{\sum term(positif)}{\sum(\text{positif})} = \frac{1}{15} = 0.067$$

b. Probabilitas kata “maaf”

$$P(maaf|negatif) = \frac{\sum term(negatif)}{\sum(\text{negatif})} = \frac{1}{4} = 0.25$$

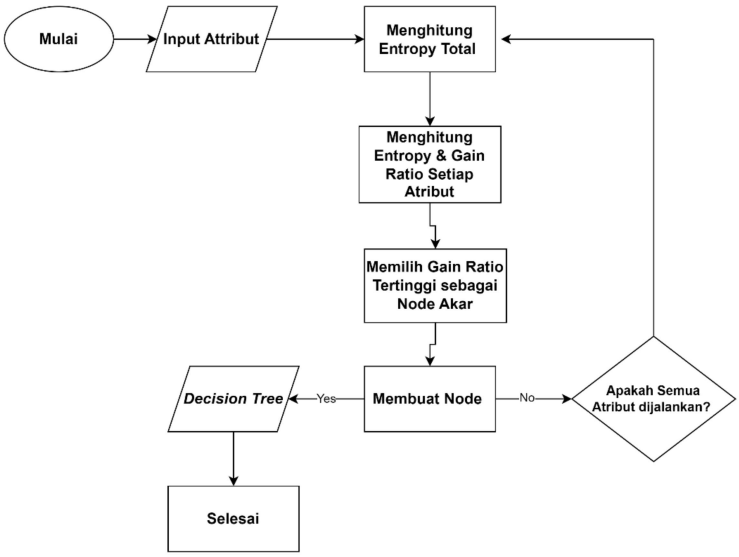
3. Proses Klasifikasi data uji didapatkan dengan mengkali semua nilai peluang. Nilai yang lebih tinggi merupakan kelas baru dari data tersebut. Sebagai contoh terdapat data testing “bagus sesuai pesan tidak kecewa”. Kata yang ada pada data latih yaitu kata “bagus dan tidak”.

$$P(Uji|positif) = P(positif)xP(bagus|positif)xP(tidak|positif)$$
$$= 0{,}5\ x\ 0{,}067\ x\ 0{,}067$$
$$= \ 0{,}00224$$
$$P(Uji|negatif) = P(negatif)xP(bagus|negatif)xP(tidak|negatif)$$
$$= 0{,}5\ x\ 0\ x\ 0{,}25$$
$$= \ 0$$

Berdasarkan perhitungan naive bayes di atas nilai tertinggi didapatkan pada kelas positif, sehingga kalimat “bagus sesuai pesan tidak kecewa” merupakan sentimen positif.

4. Decision Tree

Data hasil vektorisasi selanjutnya pengklasifikasian menggunakan model *Decision Tree*. Data akan dipartisi menjadi dua bagian, yakni data pelatihan dan data uji, dengan perbandingan 80% untuk data pelatihan dan 20% untuk data uji. *Decision tree* akan mengubah data ke dalam bentuk struktur pohon keputusan, kemudian mengubah pohon menjadi aturan dan menyederhanakan aturan tersebut.



Gambar 3. 7 Alur Pembuatan Pohon Keputusan

Berdasarkan gambar 3.7 semua kata atau atribut akan dilakukan perhitungan *entropy* dan *gain* dalam menentukan node hingga menjadi pohon keputusan. Data training yang digunakan pada tabel 3.14 adalah contoh data training yang dengan kelas sentimen negatif dan positif.

Tabel 3. 14 Sampel Data Training

Dokumen 1	bagus banget kurir juga ramah cantik tawa bosan tapi cuma sekian tidak apa apa pokok recommended	Positif
Dokumen 2	maaf barang tidak terima	Negatif
Dokumen 3	Barang sampai dengan selamat tidak rusak kualitas bagus mantap	Positif
Dokumen 4	Produk keadaan rusak toko maaf semoga toko lebih bagus lagi	Negatif

Data teks tersebut akan dimasukan kedalam sebuah tabel dengan nilai atribut 0 dan 1. Nilai atribut 1 merepresentasikan munculnya kata tersebut pada data teks, sedangkan atribut 0 mengartikan tidak munculnya kata tersebut pada data. Tabel 3.15 merupakan contoh data training yang akan dihitung *entropy* dan *gain* untuk membuat pohon keputusan.

Tabel 3. 15 Data Decision Tree

D	bagus	ramah	maaf	tidak	cantik	rusak	Kelas
D1	1	1	0	1	1	0	+
D2	0	0	1	1	0	0	-
D3	1	0	0	0	0	1	+
D4	1	0	1	0	0	1	-

Perhitungan entropy:

$$Entropy\ (total) = \left(-\tfrac{4}{4}\right) log_2 \left(\tfrac{4}{4}\right) + \left(-\tfrac{2}{4}\right) log_2 \left(\tfrac{2}{4}\right) + \left(-\tfrac{2}{4}\right) log_2 \left(\tfrac{2}{4}\right) + \left(-\tfrac{3}{4}\right) log_2 \left(\tfrac{3}{4}\right) = 1.311$$

Setelah menghitung entropy total, maka selanjutnya menghitung entropy setiap kata untuk menentukan node akar:

1. Kata “bagus”

$$Entropy\ (1) = \left(-\tfrac{2}{3}\right) log_2 \left(\tfrac{2}{3}\right) + \left(-\tfrac{1}{3}\right) log_2 \left(\tfrac{1}{3}\right) = 0.918$$

$$Entropy\ (0) = \left(-\tfrac{0}{1}\right) log_2 \left(\tfrac{0}{1}\right) + \left(-\tfrac{1}{1}\right) log_2 \left(\tfrac{1}{1}\right) = 0$$

$$Gain(total, bagus) = (1.311) - \left(\tfrac{3}{4}\right) x\ 0.918 - \left(\tfrac{1}{4}\right) x\ 0 = 0.623$$

2. Kata “ramah”

$$Entropy\ (1) = \left(-\frac{1}{1}\right) log_2 \left(\frac{1}{1}\right) + \left(-\frac{0}{1}\right) log_2 \left(\frac{0}{1}\right) = 0$$

$$Entropy\ (0) = \left(-\frac{1}{3}\right) log_2 \left(\frac{1}{3}\right) + \left(-\frac{2}{3}\right) log_2 \left(\frac{2}{3}\right) = 0.918$$

$$Gain(total, ramah) = (1.311) - \left(\tfrac{1}{4}\right) x\ 0 - \left(\tfrac{3}{4}\right) x\ 0.918 = 0.623$$

3. Kata “maaf”

$$Entropy\ (1) = \left(-\frac{0}{2}\right) log_2 \left(\frac{0}{2}\right) + \left(-\frac{2}{2}\right) log_2 \left(\frac{2}{2}\right) = 0$$

$$Entropy\ (0) = \left(-\frac{2}{2}\right) log_2 \left(\frac{2}{2}\right) + \left(-\frac{0}{2}\right) log_2 \left(\frac{0}{2}\right) = 0$$

$$Gain(total, maaf) = (1.311) - \left(\frac{2}{4}\right) x\ 0 - \left(\frac{2}{4}\right) x\ 0 = 1.311$$

4. Kata “tidak”

$$Entropy\ (1) = \left(-\frac{1}{2}\right) log_2 \left(\frac{1}{2}\right) + \left(-\frac{1}{2}\right) log_2 \left(\frac{1}{2}\right) = 1$$

$$Entropy\ (0) = \left(-\frac{1}{2}\right) log_2 \left(\frac{1}{2}\right) + \left(-\frac{1}{2}\right) log_2 \left(\frac{1}{2}\right) = 1$$

$$Gain(total, tidak) = (1.311) - \left(\frac{2}{4}\right) x\ 1 - \left(\frac{2}{4}\right) x\ 1 = 0.311$$

5. Kata “cantik”

$$Entropy\ (1) = \left(-\frac{1}{1}\right) log_2\left(\frac{1}{1}\right) + \left(-\frac{0}{1}\right) log_2\left(\frac{0}{1}\right) = 0$$

$$Entropy\ (0) = \left(-\frac{1}{3}\right) log_2\left(\frac{1}{3}\right) + \left(-\frac{2}{3}\right) log_2\left(\frac{3}{3}\right) = 0.918$$

$$Gain(total, cantik) = (1.311) - \left(\frac{1}{4}\right) x\ 0 - \left(\frac{3}{4}\right) x\ 0.918 = 0.623$$

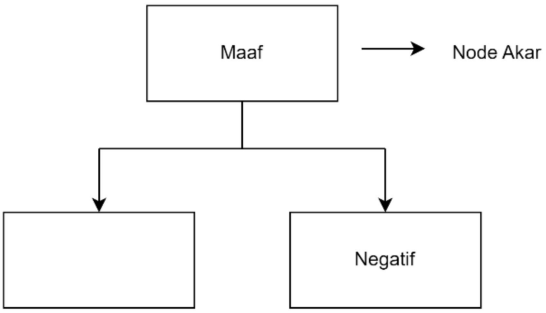
6. Kata “rusak”

$$Entropy\ (1) = \left(-\frac{1}{2}\right) log_2\left(\frac{1}{2}\right) + \left(-\frac{1}{2}\right) log_2\left(\frac{1}{2}\right) = 1$$

$$Entropy\ (0) = \left(-\frac{1}{2}\right) log_2\left(\frac{1}{2}\right) + \left(-\frac{1}{2}\right) log_2\left(\frac{1}{2}\right) = 1$$

$$Gain(total, rusak) = (1.311) - \left(\frac{2}{4}\right) x\ 1 - \left(\frac{2}{4}\right) x\ 1 = 0.311$$

Dari hasil *entropy* dan *gain* semua *term* (kata) maka didapatkan nilai dengan *gain* tertinggi pada kata maaf dengan nilai *gain 1.311. Rule* (ketentuan) pohon dapat dilihat pada gambar 3.8



Gambar 3. 8 Node Akar

Gambar 3.8 Menggambarkan node akar pada pohon keputusan yang terpilih yaitu kata “maaf”. Untuk memilih sub-node pada pohon dilakukan dengan menghitung *information gain* kembali setiap fitur yang belum digunakan sebagai node dalam subset tersebut dan memilih *gain ratio* tertinggi sebagai sub node terpilih, ulangi langkah tersebut sampai hingga mencapai kondisi berhenti atau batas kedalaman pohon.

3.3.10 Evaluasi dan Analisis

Setelah model terbentuk dari masing – masing metode, maka langkah selanjutnya yaitu mengevaluasi kinerja kedua model tersebut. Dalam proses

pengujian ini, *confusion matrix* digunakan untuk menganalisis hasil prediksi model dengan melihat hasil akurasi dari kedua model dan membandingkan hasil akurasi ketiga model berdasarkan skenario yang ada yaitu penerapan kamus *slang word* dan tanpa penerapan korpus *slang word.*