BAB III

METODOLOGI DAN PERANCANGAN SISTEM

3.1 Metodologi Penelitian

Metodologi penelitian yang digunakan dalam penelitian ini adalah sebagai berikut.

1. Wawancara

   Wawancara dilakukan untuk mengetahui gambaran permasalahan yang ada dan tingkat kebutuhan sistem yang akan dibangun untuk menyelesaikan permasalahan yang ada, dalam hal ini mengklasifikasi artikel ke dalam kategori *personal growth* untuk dimasukkan ke dalam aplikasi *mobile* MyValue. Wawancara dilakukan secara langsung dengan *product manager* tim OVAL PT Kompas Gramedia.

2. Studi Literatur

   Dalam tahap ini, dilakukan pembelajaran dan pendalaman teori-teori terkait pengambilan sampel, *text mining*, algoritma Multinomial Naïve Bayes, *confusion matrix*, dan MyValue. Sumber informasi yang didapatkan berasal dari penelitian terdahulu, buku, jurnal, referensi, *website* ataupun data yang terkait dengan penelitian yang akan dilakukan.

3. Perancangan Sistem

   Secara garis besar, sistem menampilkan daftar artikel yang termasuk dalam kategori *personal growth* ke dalam aplikasi *mobile* MyValue. Untuk mencapai hasil ini beberapa proses yang perlu dilakukan adalah sebagai berikut.

a. *Text-preprocessing* (proses pengubahan bentuk data yang belum terstruktur menjadi data yang terstruktur), proses ini terdiri dari proses sebagai berikut.
    i. HTML *stripping* (menghilangkan *tag* HTML pada dokumen) menggunakan library Python BeautifulSoup4.
    ii. *Case folding* (mengubah semua huruf dalam dokumen) menggunakan *function* bawaan Python yaitu lower.
    iii. *Tokenizing* (menghilangkan simbol dan tanda baca) menggunakan *function* bawaan Python yaitu translate.
    iv. *Filtering* (menghilangkan kata-kata yang tidak penting dengan *stopwords*) menggunakan fungsi yang disediakan oleh *library* Sastrawi yaitu remove.
    v. *Stemming* (mengambil kata dasar dari sebuah kata) menggunakan fungsi yang disediakan oleh *library* Sastrawi stem.
b. *Modeling* (menghitung jumlah kemunculan kata dalam tiap artikel berdasarkan *vocabulary*) dalam bentuk *bag of words*.
c. *Training* (menghitung *conditional probability* yang dimiliki oleh setiap kata).
d. *Validating* (proses menguji hasil pelatihan).
e. Klasifikasi (mengkategorikan suatu artikel termasuk dalam kategori *personal growth* atau *non personal growth*) menggunakan kerangka kerja Flask untuk menyediakan antarmuka (API) untuk di-*consume* oleh aplikasi *mobile* MyValue.

Setelah didapatkan informasi yang dibutuhkan, maka dapat dibuat rancangan sistem yang akan dibangun. Perancangan dimulai dengan membuat gambaran umum sistem, perancangan *flowchart*, perancangan struktur tabel, dan perancangan antarmuka aplikasi mobile MyValue untuk hasil implementasi Naïve Bayes.

4. Implementasi

   Pada tahap ini dilakukan proses implementasi dari rancangan yang telah disusun pada tahap-tahap sebelumnya. Berbagai proses yang dibutuhkan dikerjakan pada tahap ini menggunakan bahasa pemrograman Python. Pada bagian klasifikasi, digunakan kerangka kerja Flask untuk menyediakan sebuah antarmuka (API) bagi aplikasi *mobile* MyValue.

5. Pengujian dan Evaluasi

   Setelah sistem selesai dibangun, dilakukan pengujian terhadap implementasi algoritma Multinomial Naïve Bayes dalam mengklasifikasikan artikel pada aplikasi *mobile* MyValue. Data yang digunakan dalam tahap pengujian merupakan data artikel yang sudah diberi label oleh ahli dari tim OVAL PT Kompas Gramedia. Pengujian dilakukan dengan melakukan validasi dan menghitung *f-measure*, akurasi, *precision*, dan *recall* dari algoritma Multinomial Naïve Bayes dalam mengklasifikasi artikel pada aplikasi *mobile* MyValue.

6. Dokumentasi

   Pada tahap ini, dibuat sebuah laporan sebagai dokumentasi dari penelitian dan pembuatan aplikasi secara bertahap, mulai dari pendahuluan hingga kesimpulan dan saran.

3.2 Teknik Pengumpulan Data

Pengumpulan data untuk mengetahui permasalahan yang ada dan kebutuhan PT Kompas Gramedia akan sistem yang dapat mengklasifikasikan artikel ke dalam kategori *personal growth* dilakukan dengan cara wawancara dengan *product manager* tim OVAL PT Kompas Gramedia, Adya Danaditya.

Data yang dibutuhkan untuk tahap *training* dan *validation* adalah artikel yang sudah diberi label oleh tim *expert* OVAL PT Kompas Gramedia. Untuk artikel yang sudah termasuk dalam kategori *personal growth* diambil langsung dari API MyValue dan diberi label *personal growth*. Untuk data yang belum dikategorikan ke dalam kategori *personal growth* atau *non personal growth* diambil dari NewsAPI, kemudian diberi label secara manual. 70% dari data yang telah diklasifikasi secara manual akan digunakan untuk *training* dan perhitungan kemunculan masing-masing kata. Sedangkan 30% dari data tersebut akan digunakan untuk validasi yang akan menentukan *f-score*, akurasi, *precision*, dan *recall*. Detail dari kumpulan data set yang didapat dapat dilihat pada Tabel 3.1.

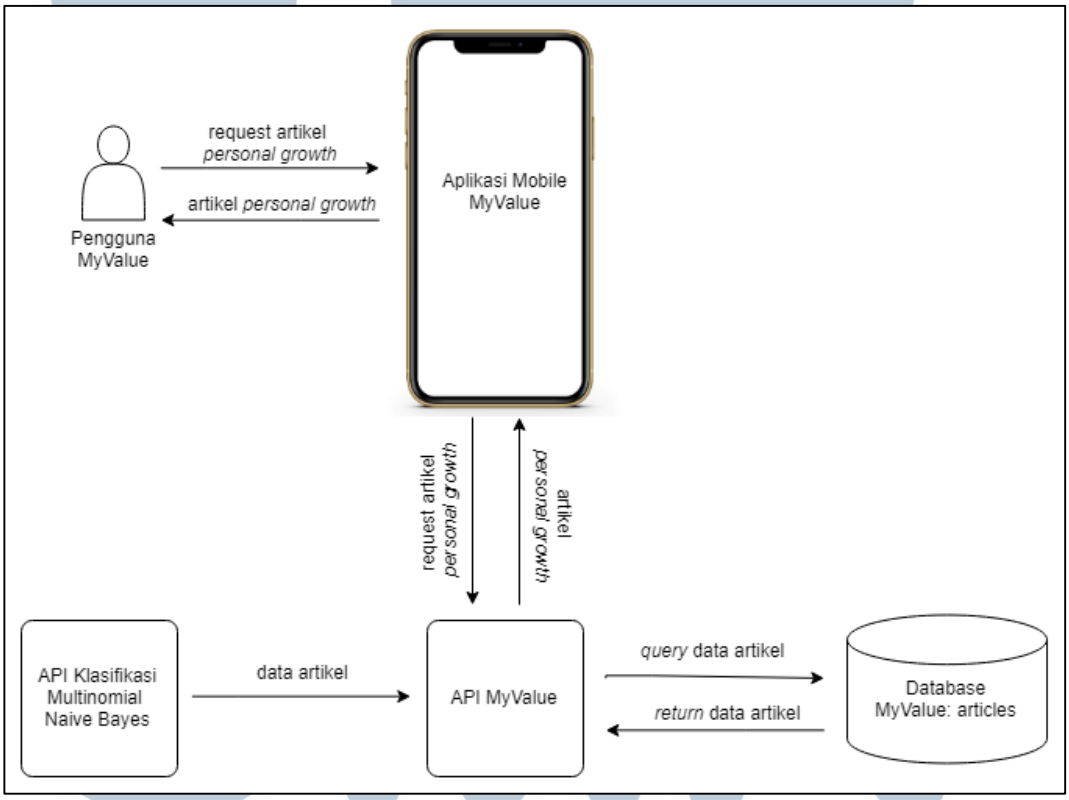
Tabel 3.1 Detai Data Set

Kategori	**Jumlah**	
	Data Training	**Data Validasi**
Personal Growth	35	15
Non Personal Growth	35	15
Total	70	30

Komposisi jumlah data *training* dan validasi masing-masing kategori dijadikan variabel yang memengaruhi (variabel bebas) terhadap *f-score*, akurasi, *precision*, dan *recall* (variabel terikat). Selanjutnya akan dibahas lebih lanjut pada skenario pengujian di Bab 4.

3.3 Gambaran Umum Sistem

Sistem yang dibangun merupakan *Application Programming Interface* (API) yang diimplementasikan menggunakan Python Flask. Sistem ini berfungsi untuk mengklasifikasikan artikel ke dalam kategori *personal growth* dan *non personal growth*. Selanjutnya kedua klasifikasi ini akan disebut sebagai "*personal growth*" untuk artikel yang termasuk dalam kategori *personal growth* dan "*non personal growth*" untuk artikel yang termasuk dalam kategori *non personal growth*. Model sistem digambarkan pada Gambar 3.1.



Gambar 3.1 Gambaran Umum Sistem Klasifikasi Artikel

Pengguna MyValue membuka aplikasi *mobile* MyValue untuk melihat daftar artikel *personal growth*. Dalam sistem yang sudah ada sekarang, *user* akan mengirim *request* ke aplikasi *mobile* MyValue, aplikasi *mobile* MyValue akan meneruskan *request* tersebut ke API MyValue. API MyValue akan menarik data

artikel dari *database*, lalu dikembalikan sesuai dengan *flow* sebelumnya. Yang disisipkan ke dalam sistem dalam penelitian ini adalah API klasifikasi menggunakan Multinomial Naïve Bayes, sehingga sebelum menarik data dari *database*, API klasifikasi akan mengirimkan artikel *personal growth* hasil klasifikasi dengan Multinomial Naïve Bayes.

Pembuatan API klasifikasi menggunakan Multinomial Naïve Bayes diawali dengan pengumpulan data *training* berupa artikel yang telah diklasifikasikan menjadi kategori "*personal growth*" dan kategori "*non personal growth*". Pengklasifikasian artikel dilakukan oleh tim *expert* dari OVAL PT Kompas Gramedia.

Kemudian akan dilakukan proses *text preprocessing* pada data artikel. Sebelum dilakukan *text preprocessing*, dilakukan HTML *stripping* terhadap data untuk menghilangkan *tag* HTML. Data artikel yang diambil dari API masih dalam bentuk mentah, sehingga perlu dibersihkan terlebih dahulu. Di tahap *text preprocessing* ada empat tahap yang dilakukan, yaitu *case folding* (mengubah dokumen menjadi huruf kecil), *tokenization* (menghilangkan symbol dan tanda baca), *filtering* (menghilangkan kata-kata yang tidak penting, dan *stemming*. Sesuai dengan batasan masalah, maka tahapan ini dianggap cukup untuk pemrosesan kata-kata yang formal.

Secara garis besar, proses *text preprocessing* yang dilakukan adalah pengubahan dokumen menjadi huruf kecil (*case folding*), menghilangkan simbol dan tanda baca (*tokenizing*), menghilangkan kata-kata yang tidak penting (*filtering)*, dan mengubah kata ke dalam bentuk dasarnya (*stemming*). Proses *case folding* dikerjakan menggunakan fungsi lower yang merupakan bawaan dari Python. Proses

tokenizing dikerjakan menggunakan fungsi translate yang juga merupakan bawaan dari Python. Sementara untuk proses *filtering* dan *stemming* digunakan *library* tambahan yaitu Sastrawi. Pada proses *filtering*, Sastrawi sudah menyediakan *stopwords* dalam *class* StopWordRemoverFactory, sehingga tidak perlu lagi mencari *stopwords* lainnya. Sama pula halnya dengan proses *stemming*, Sastrawi menyediakan proses *stemming* dalam class StemmerFactory, sehingga kata-kata dari hasil *tokenizing* dapat dengan mudah diubah menjadi bentuk dasarnya.

Setelah dilakukan *text preprocessing*, masing-masing kata dalam tiap kategori akan dihitung perhitungan kemunculan kata dan peluang tiap kata pada masing-masing kategori. Jumlah kemunculan kata dan peluang tiap kata pada masing-masing kategori akan disimpan dalam *file*.

Hasil *training* menjadi panduan bagi sistem untuk mengklasifikasikan artikel yang termasuk *personal growth* dan yang bukan termasuk *personal growth*. Aplikasi *mobile* MyValue perlu meng-*consume* API ini ketika hendak menggunakannya. API akan melakukan serangkaian proses yang telah dijabarkan di atas dan mengeluarkan *output* berupa artikel yang termasuk dalam kategori *personal growth*.

Saat proses validasi, dilakukan proses *text preprocessing* yang sama seperti pada proses *training*. Setelah dilakukan *text preprocessing*, maka dapat dilakukan perhitungan Naïve Bayes untuk menentukan kategori dari data artikel. Setiap kata pada artikel akan dipetakan ke dalam *array* yang berisi jumlah kemunculan kata tersebut apabila memiliki kategori "*personal growth*" dan kategori "*non personal growth*". Setelah dilakukan perhitungan dengan rumus Naïve Bayes, dilakukan pengkategorian artikel. Pengkategorian artikel dilakukan dengan mengambil hasil

perhitungan tertinggi antara kedua kategori. Hasil tersebut disimpan dalam *database* untuk kemudian digunakan oleh aplikasi *mobile* MyValue.

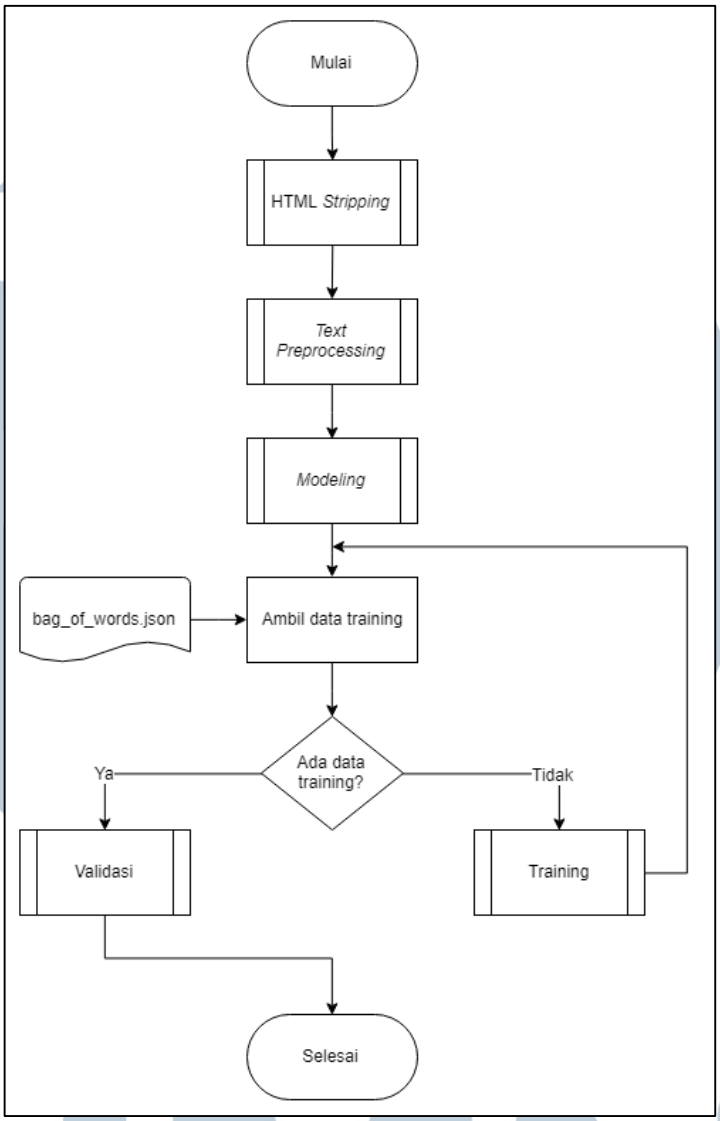
Data yang dikirim ke aplikasi *mobile* MyValue berupa daftar artikel yang termasuk dalam kategori *personal growth*. Hasil ini dapat diakses oleh pengguna aplikasi *mobile* MyValue saat mengakses bagian artikel pada aplikasi *mobile* MyValue.

3.4 Perancangan Sistem

Secara umum, sistem yang dibangun terdiri dari dua aplikasi yang berbeda. Aplikasi pertama adalah aplikasi *training* dan validasi yang dibuat dengan bahasa pemrograman Python. Keluaran dari aplikasi ini adalah hasil *training* berupa *bag of words* dalam bentuk JSON. Hasil *training* digunakan untuk acuan pada proses validasi. Aplikasi kedua adalah *article classifier* yang dibuat menggunakan kerangka kerja Flask dan bahasa pemrograman Python. API *article classifier* digunakan untuk proses pengklasifikasian artikel baru. Untuk kepentingan demo aplikasi, dibuat pula *sample* API MyValue dan *sample* aplikasi *mobile* MyValue. *Sample* API MyValue memasukkan artikel yang sudah dikategorikan sebagai *personal growth* ke dalam *sample* aplikasi *mobile* MyValue. Perancangan sistem ini terdiri dari *flowchart* untuk menjelaskan alur API yang dibuat dan antarmuka aplikasi *mobile* MyValue untuk memperlihatkan hasil klasifikasi.

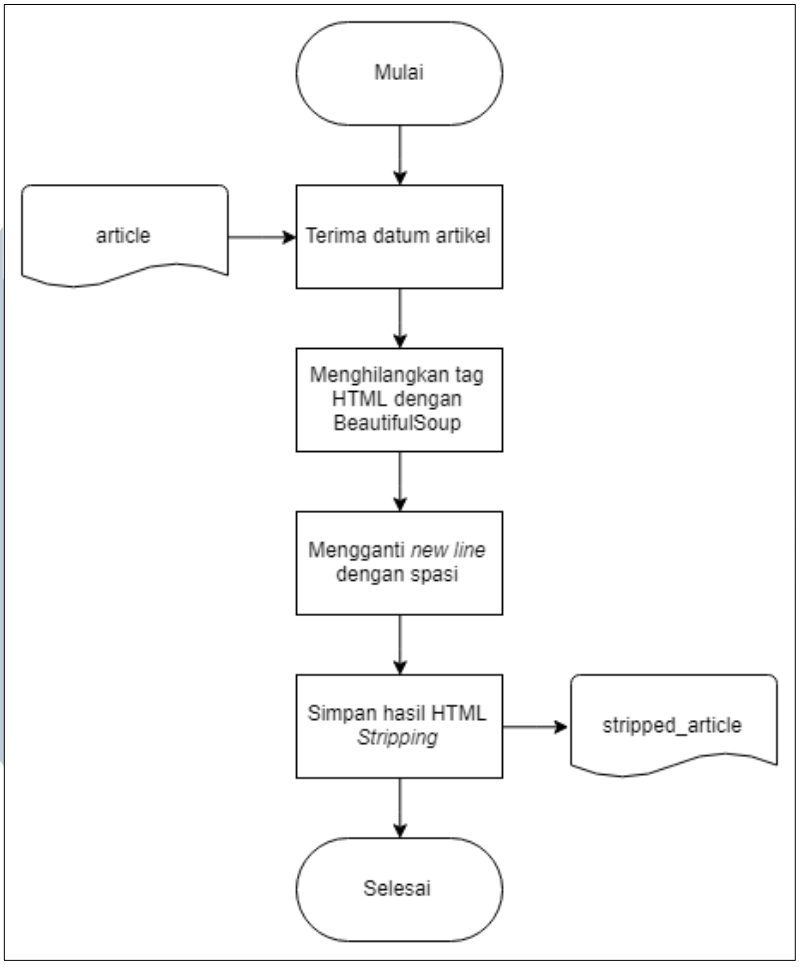
3.1.1 Flowchart Aplikasi Training dan Validasi

Gambar 3.2 merupakan alur aplikasi *training* dan validasi sistem klasifikasi artikel dengan algoritma Multinomial Naïve Bayes. Aplikasi *training* dan *validasi* adalah inti dari implementasi algoritma Multinomial Naïve Bayes untuk klasifikasi artikel pada aplikasi MyValue.



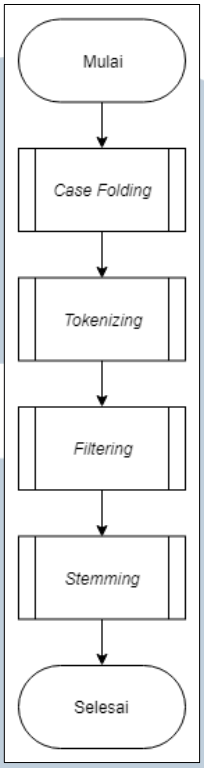
Gambar 3.2 *Flowchart* Aplikasi *Training* dan Validasi

Pada awal aplikasi, saat artikel yang ingin diklasifikasi masuk ke dalam sistem, dilakukan proses HTML *Stripping* untuk menghilangkan *tag* HTML dan elemen *new line* "\n" dari artikel. Hasil HTLM *Stripping* disimpan dalam sebuah file berformat json. Proses HTML *Stripping* ditunjukkan pada Gambar 3.3.



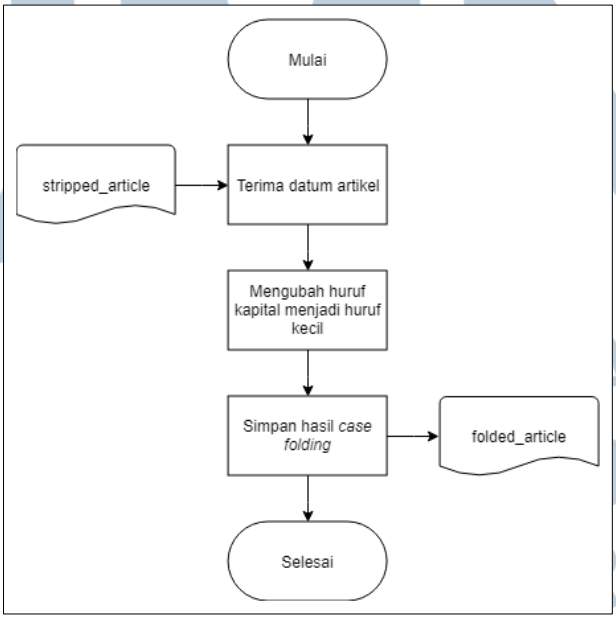
Gambar 3.3 *Flowchart* Modul HTML *Stripping*

Dalam modul *text preprocessing*, ada empat modul besar, dimana masing-masing modul memiliki peranan pentingnya masing-masing. Modul-modul dalam proses *text preprocessing* antara lain *case folding*, *tokenizing*, *filtering*, dan *stemming*. Proses *text preprocessing* ditunjukkan pada Gambar 3.4.



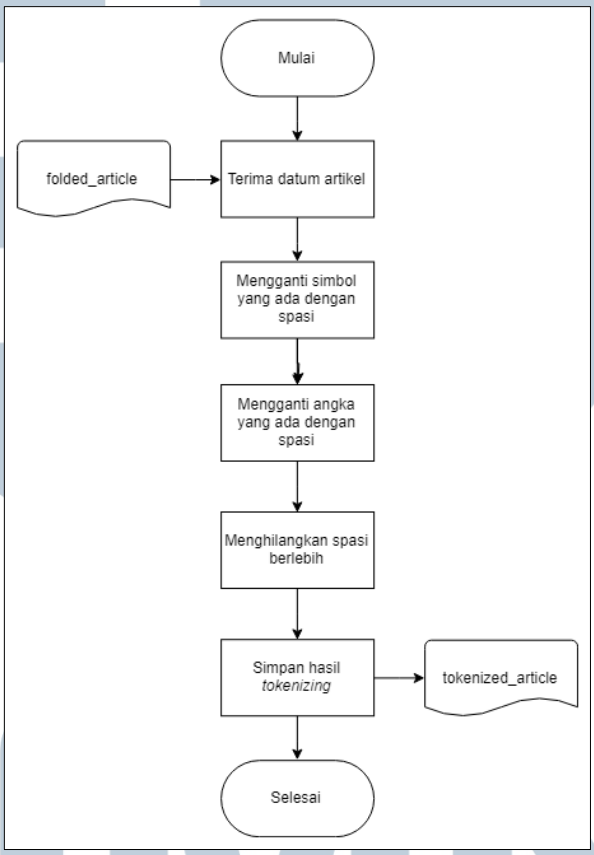
Gambar 3.4 *Flowchart* Modul *Text Preprocessing*

Pada modul *case folding*, dilakukan proses transformasi data artikel yang sudah melewati proses HTML *stripping* menjadi huruf kecil semua. Proses ini dilakukan dengan menggunakan fungsi *lower* dari Python. Proses *case folding* ditunjukkan pada Gambar 3.5.



Gambar 3.5 *Flowchart* Modul *Case Folding*

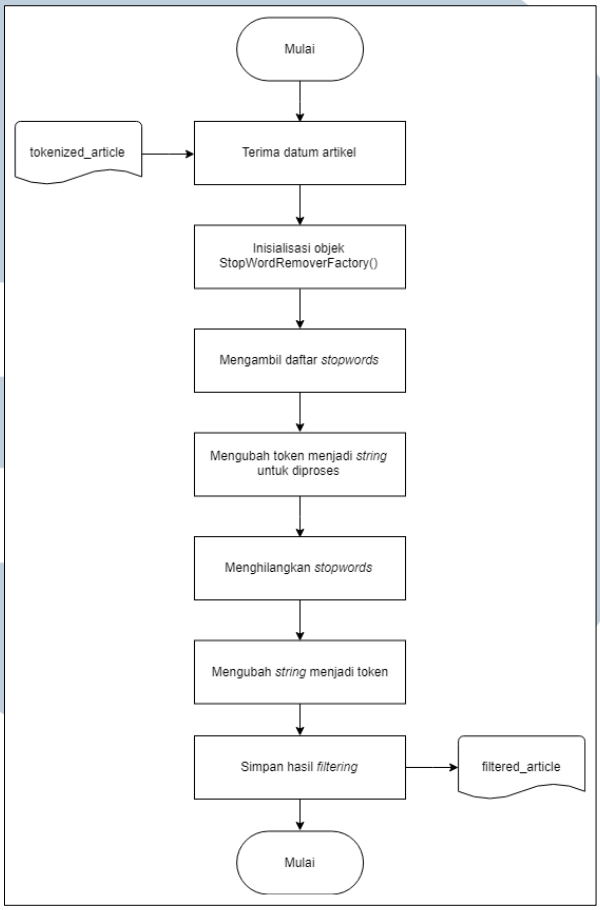
Selanjutnya, pada modul *tokenizing*, dilakukan penghapusan elemen-elemen tidak penting seperti tanda baca dan angka. Proses ini dilakukan dengan cara mengganti setiap tanda baca dan angka dengan spasi kemudian menghapus spasi berlebih dengan fungsi *trim* pada Python. Proses *tokenizing* ditunjukkan pada Gambar 3.6.



Gambar 3.6 *Flowchart* Modul *Tokenizing*

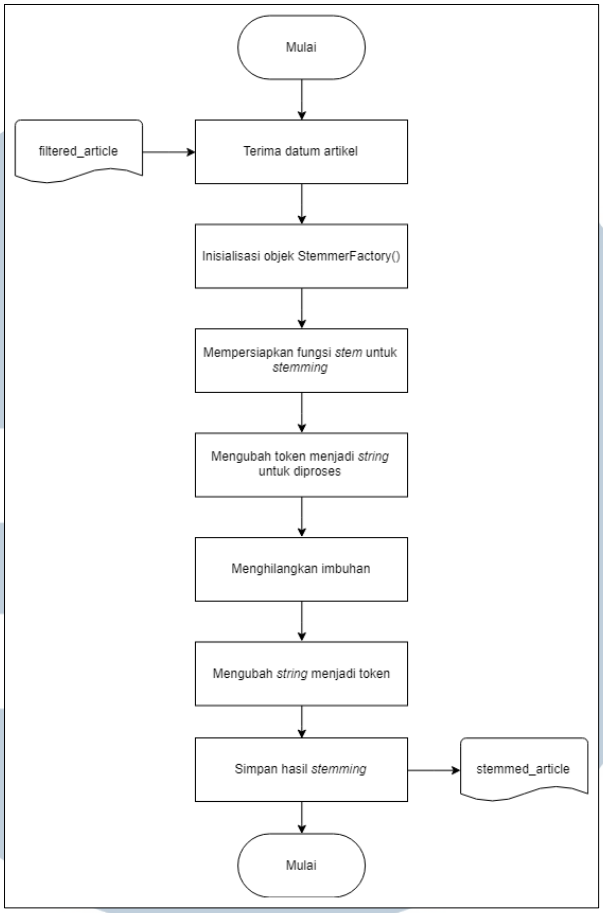
Setelah proses *tokenizing* berhasil dilewati, dilanjutkan dengan proses *filtering*. Proses *filtering* dilakukan dengan bantuan kamus *stopword* dari *library* Sastrawi. Pertama, akan diinisialisasi objek StopWordRemoverFactory(), kemudian *stopwords* diambil. Sebelum dibersihkan, token dari proses *tokenizing* harus diubah menjadi *string* sementara karena parameter yang diterima pada proses *filtering* ini adalah *string*. Setelah dijadikan *string*, kata-kata yang tidak penting

dihapus dan diubah menjadi token kembali untuk digunakan pada proses selanjutnya. Proses *filtering* ditunjukkan pada Gambar 3.7.



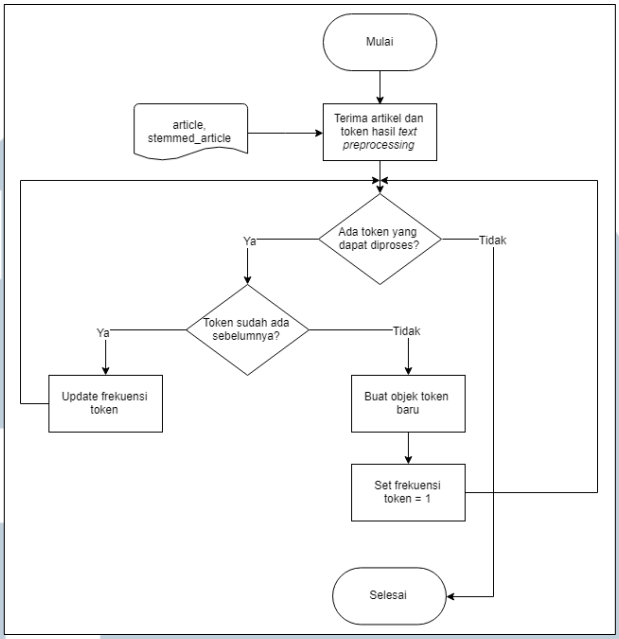
Gambar 3.7 *Flowchart* Modul *Filtering*

Langkah terakhir dari proses *text preprocessing* adalah *stemming*. *Stemming* adalah proses mengubah token ke dalam bentuknya yang paling dasar. Proses ini dibantu oleh *library* Sastrawi. Pertama, akan diinisialisasi objek StemmerFactory(), kemudian akan dipersiapkan untuk melakukan proses *stemming*. Sama seperti proses *filtering*, token harus diubah menjadi *string* sementara karena parameter yang diterima pada proses *stemming* adalah *string*. Setelah dijadikan *string*, proses *stemming* dilakukan. Setelah itu barulah diubah menjadi bentuk token kembali dan disimpan untuk digunakan pada proses selanjutnya. Proses *stemming* ditunjukkan pada Gambar 3.8.



Gambar 3.8 *Flowchart* Modul *Stemming*

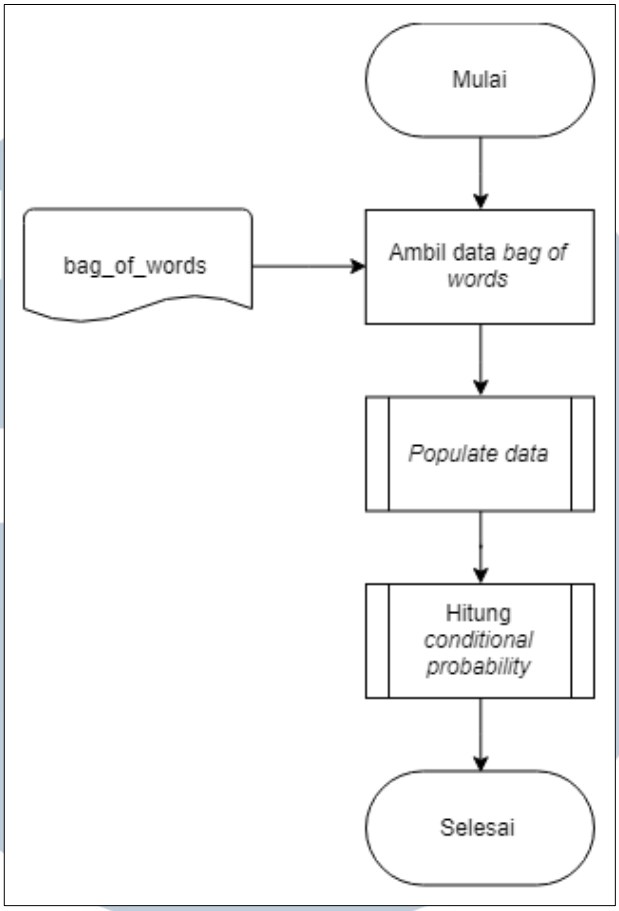
Setelah proses *text preprocessing* selesai, tahap selanjutnya adalah *modeling*. Tujuan dibuatnya model adalah untuk menghitung jumlah kemunculan kata. Model dibuat dalam bentuk *bag of words*. Dalam tahap ini dibuat dua *bag of words*, satu untuk masing-masing kategori. Semua kata yang termasuk dalam kategori *personal growth* akan masuk ke dalam *bag of words personal growth* dan kata yang termasuk dalam kategori *non personal growth* akan masuk ke dalam *bag of words non personal growth*. Selain kata, dihitung juga jumlah kemunculan setiap kata dalam kelas tersebut. Proses *modeling* ditunjukkan pada Gambar 3.9.



Gambar 3.9 *Flowchart* Modul *Modeling*

Bag of words ini akan menjadi acuan dalam proses *training*. Proses *training* diawali dengan proses *populate data*. *Populate data* adalah proses untuk membagi data ke dalam data *training* dan data validasi. *Populate data* dilakukan dengan mengambil data acak tiap kali berjalan, sehingga nantinya akan menghasilkan hasil belajar yang tidak selalu sama.

Setelah *populate* data, dilakukan perhitungan peluang kemunculan kata yang akan disimpan dalam *file bag of words* sebagai *conditional probability*. Peluang kemunculan kata dihitung dengan rumus *conditional probability* dengan *Laplace's law of smoothing*, yaitu menjumlahkan kemunculan kata[ctr] untuk setiap kategori dengan satu, kemudian dibagi dengan penjumlahan total kata untuk kategori tersebut dan jumlah *vocabulary*. Proses *training* ditunjukkan pada Gambar 3.10.

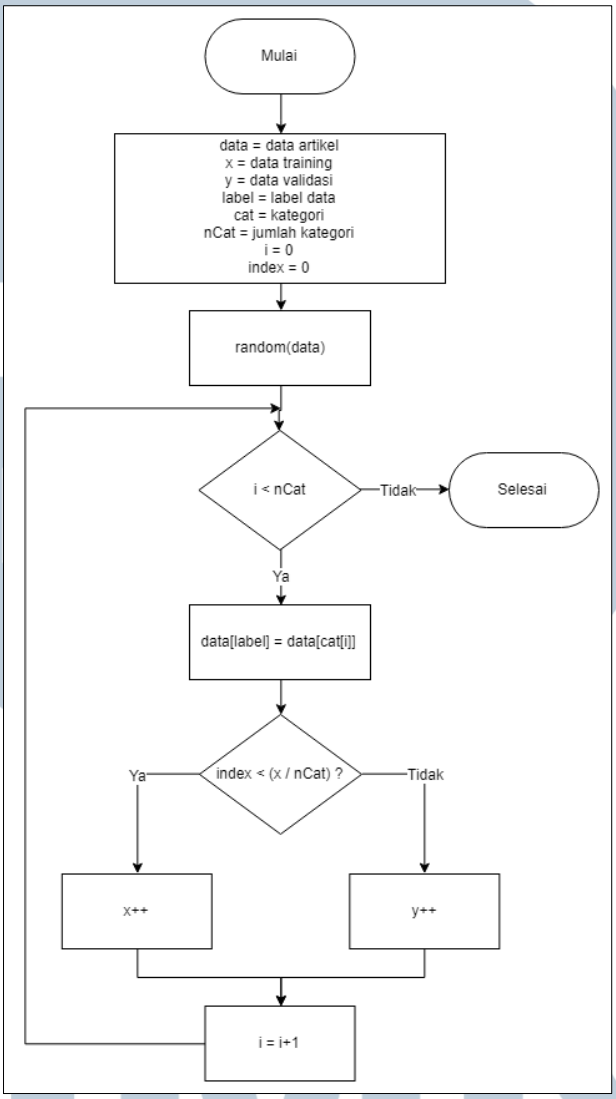


Gambar 3.10 *Flowchart* Modul *Training*

Dalam modul *training*, terdapat dua modul besar, yaitu *populate* data dan hitung *conditional probability*. *Populate* data adalah modul untuk membagi data *training* dan validasi, sementara hitung *conditional probability* adalah modul untuk menghitung peluang kemunculan suatu kata *w* dalam kelas *c*. Nilai dari *conditional probability* akan digunakan untuk menghitung *likelihood* dan *posterior probability* pada tahap validasi. menghitung dan dapat dilakukan validasi.

Dalam modul *populate* data, data artikel diacak untuk mendapatkan hasil *training* terbaik. Proses *populate* data dilakukan per kategori artikel. Nilai yang diinginkan bisa diatur pada program untuk kepentingan validasi nantinya. Berdasarkan penelitian yang dilakukan Hamzah (2002), Naïve Bayes belajar paling

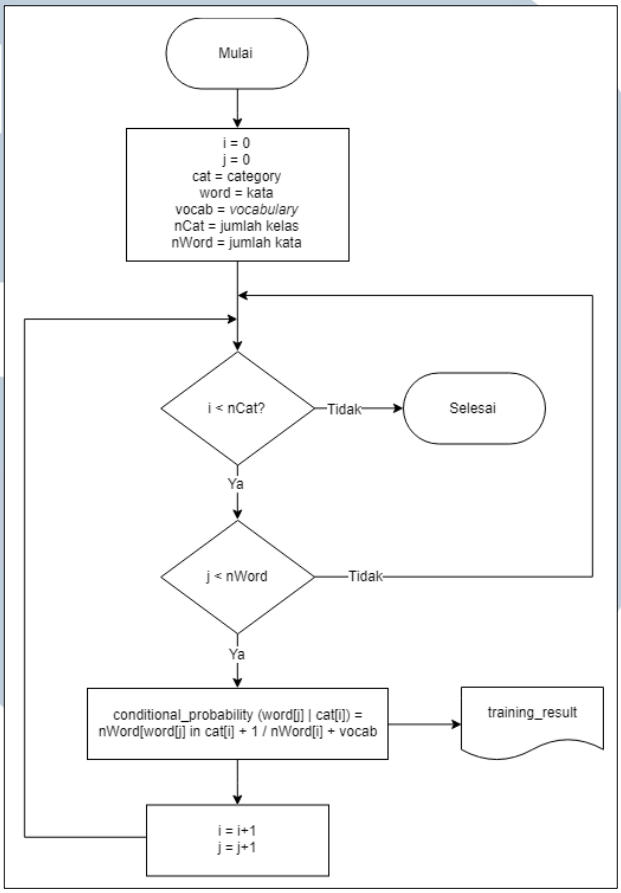
baik ketika perbandingan data *training* dan data *validasi* adalah 70 dan 30 berturut-turut. Proses *populate* data ditunjukkan pada Gambar 3.11.



Gambar 3.11 *Flowchart* Modul *Populate* Data

Modul kedua dari proses *training* adalah perhitungan *conditional probability*. *Conditional probability* menghitung peluang kemunculan suatu kata dalam dokumen *training* tertentu. Pada proses ini, karena mungkin saja ada suatu kata yang tidak muncul sama sekali dalam dokumen *training* tertentu (*out of vocabulary*) dan menghasilkan *conditional probability* 0, maka ditambahkan nilai

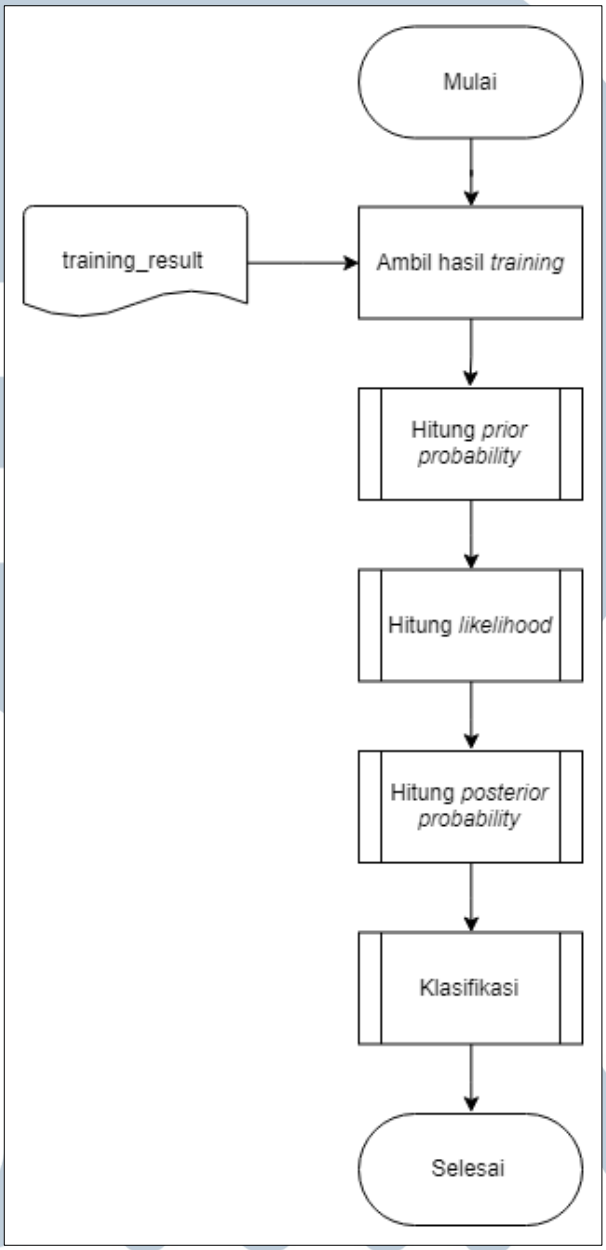
1 sesuai dengan aturan *Laplace's law of smoothing*. Proses perhitungan conditional probability ditunjukkan pada Gambar 3.12.



Gambar 3.12 *Flowchart* Modul *Conditional Probability*

Proses *training* berhenti sampai perhitungan *conditional probability* dan kemudian dilanjutkan dengan proses validasi. Proses validasi menggunakan nilai *conditional probability* dari proses *training* yang disimpan dalam *file*. Sama seperti proses *training*, proses validasi juga terdiri dari beberapa modul, di antaranya hitung *prior probability*, hitung *likelihood*, hitung *posterior probability*, dan proses klasifikasi itu sendiri. Tujuan dari validasi adalah untuk menguji hasil *training* dan mengetahui apakah model yang sudah digunakan proses *training* sudah optimal untuk kasus ini. Jika belum memadai, maka akan dilakukan *training* ulang untuk

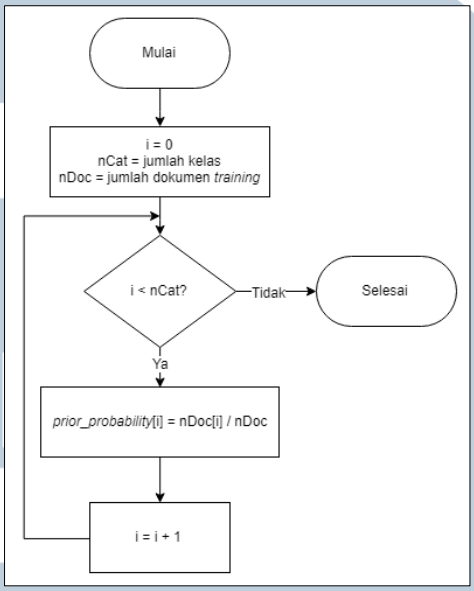
mendapatkan hasil seoptimal mungkin. Proses *training* ditunjukkan pada Gambar 3.13.



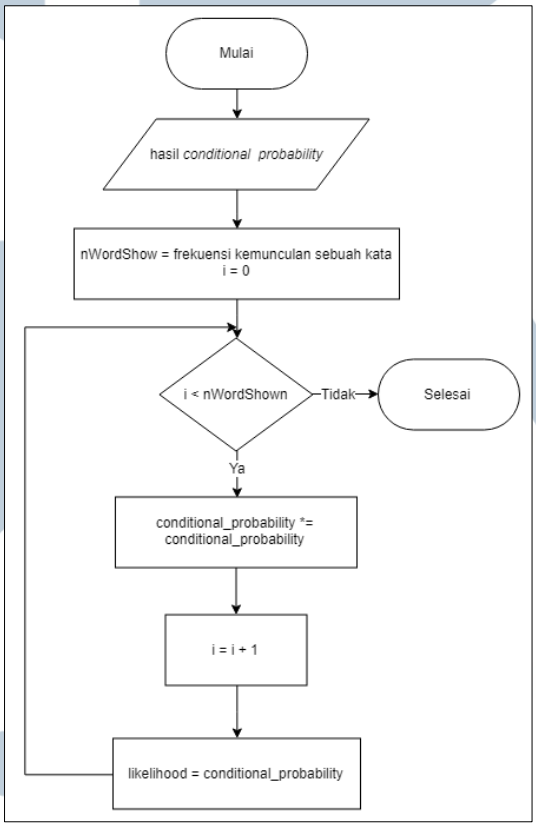
Gambar 3.13 *Flowchart* Modul *Validasi*

Prior probability adalah peluang munculnya sebuah dokumen dengan label atau kelas tertentu dalam sebuah dokumen training. Dalam kasus ini, *prior probability* diartikan sebagai peluang munculnya artikel *personal growth* dan atau

non personal growth dalam data *training*. Proses perhitungan *prior probability* ditunjukkan pada Gambar 3.14.



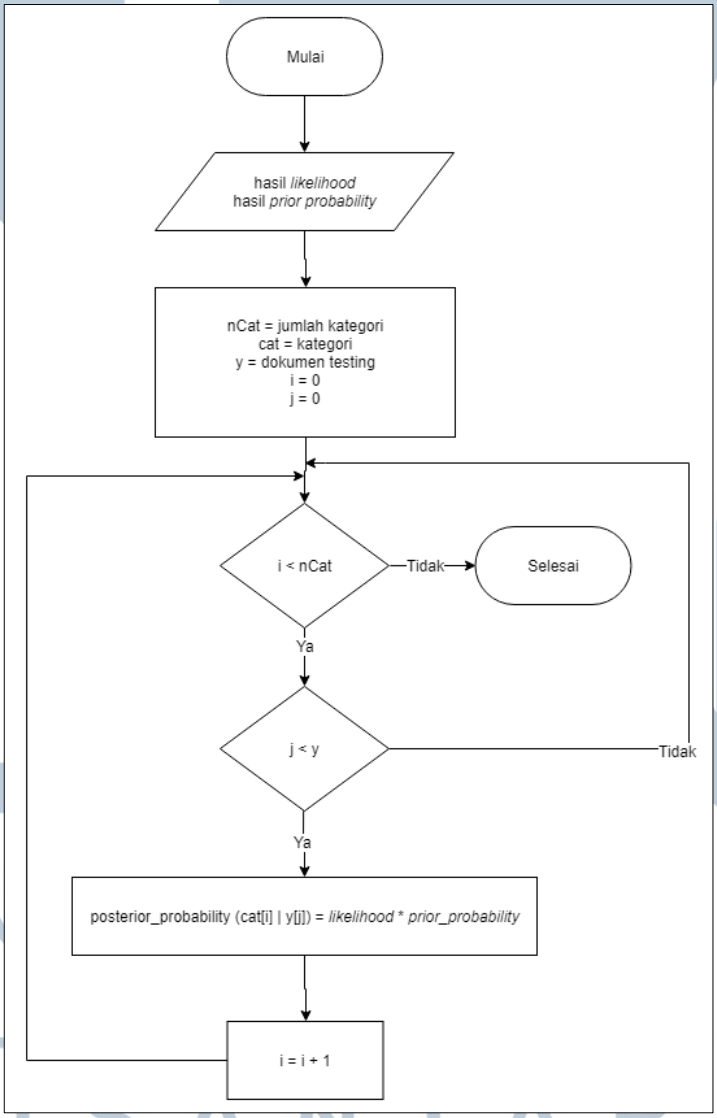
Gambar 3.14 *Flowchart* Modul *Prior Probability*



Gambar 3.15 *Flowchart* Modul *Likelihood*

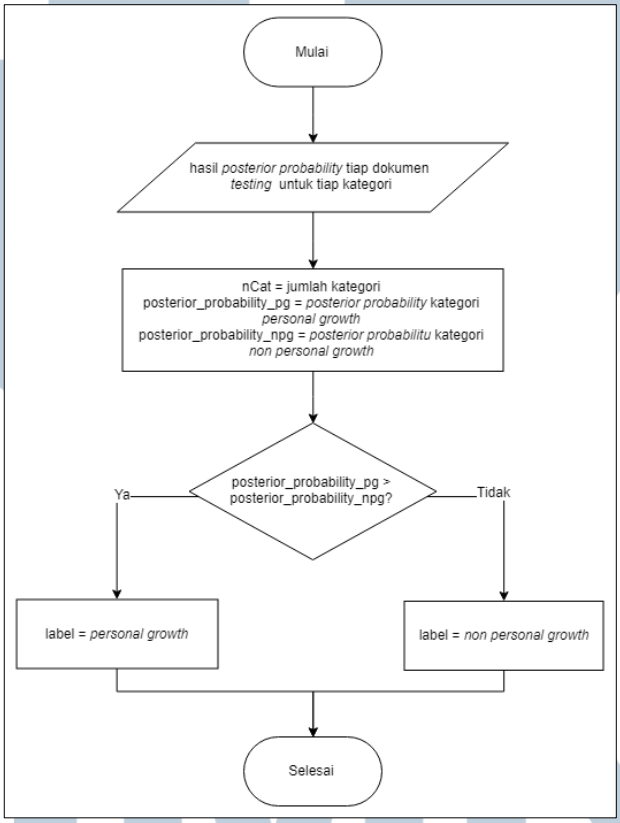
Likelihood adalah set peluang kemunculan kata dalam dokumen tertentu. Nilainya ditentukan dengan perkalian antara peluang kemunculan kata (*conditional probability*) dengan dirinya sendiri sebanyak frekuensi kemunculan kata, mengikuti pola data validasi. Proses perhitungan *likelihood* ditunjukkan pada Gambar 3.15.

Setelah *likelihood*, dihitung *posterior probability*. *Posterior probability* secara sederhana dalam kasus ini diartikan sebagai peluang suatu artikel masuk ke dalam kategori *personal growth* ataupun *non personal growth*. Proses perhitungan *posterior probability* ditunjukkan pada Gambar 3.16.



Gambar 3.16 *Flowchart* Posterior *Probability*

Tahap terakhir dalam proses validasi adalah klasifikasi. Pada dasarnya, klasifikasi hanya membandingkan nilai *posterior probability* tiap kategori untuk tiap artikel. Jika *posterior probability* kategori *personal growth* lebih besar, maka artikel akan diberi label *personal growth* dan jika *posterior probability* kategori *non personal growth* lebih besar, maka artikel akan diberi label *non personal growth*. Proses klasifikasi ditunjukkan pada Gambar 3.17.



Gambar 3.17 *Flowchart* Klasifikasi

3.1.2 Perancangan Struktur Tabel

Dalam proses pengembangan sistem klasifikasi artikel menggunakan algoritma Multinomial Naïve Bayes, diperlukan satu tabel untuk menyimpan hasil klasifikasi artikel dengan kategori *personal growth.* Basis data yang digunakan adalah PostgreSQL, sesuai dengan basis data yang digunakan oleh tim OVAL

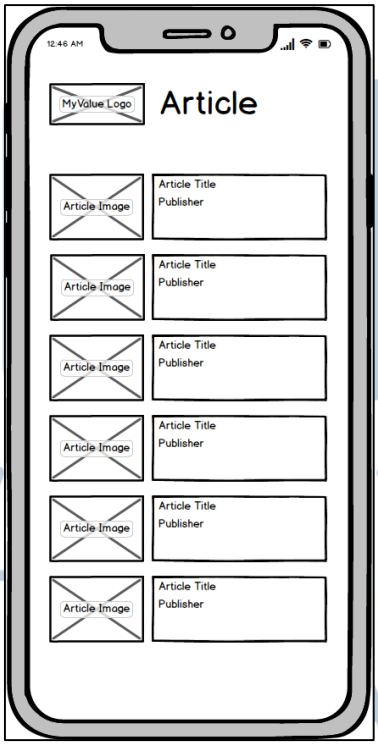
Kompas Gramedia dalam mengembangkan aplikasi *mobile* MyValue. Struktur tabel dijelaskan pada Tabel 3.2.

Tabel 3.2 Struktur Tabel Artikel

Nama Kolom	Tipe Data	Nullable	Identitas
id	int(11)		PK
title	varchar(100)	not null	
content	text	not null	
publisher	varchar(20)	not null	
image	varchar(150)	not null	

3.1.3 Perancangan Antar Muka Sistem

Inti sistem yang akan dibangun adalah *Application Programming Interface* (API) yang akan di-*consume* oleh aplikasi *mobile* MyValue. Karena akses untuk menyisipkan API klasifikasi artikel menggunakan algoritma Multinomial Naïve Bayes tidak diberikan oleh PT Kompas Gramedia, dibuatlah rancangan antarmuka untuk membantu memperlihatkan hasil klasifikasi artikel dalam aplikasi. Rancangan antarmuka halaman daftar artikel dapat dilihat pada Gambar 3.4



Gambar 3.4 Rancangan Antarmuka Halaman Artikel MyValue