BAB III

METODOLOGI PENELITIAN DAN PERANCANGAN SISTEM

3.1 Metodologi Penelitian

Metode penelitian yang digunakan dalam mengimplementasikan algoritma K-Means dan Gaussian Mixture Model untuk automasi pengelompokkan berita di kompas.com adalah sebagai berikut.

1. Studi Literatur

Pada tahap ini, dilakukan pembelajaran mengenai Berita, Media Massa, *Information Retrieval*, *Preprocessing*, *Case Folding*, *Filtering*, *Stemming*, *Term Weighting*, K–Means, GMM, PCA, & *Silhouette Coefficient*. Materi pembelajaran didapat dari berbagai sumber seperti buku, referensi, situs online, data ataupun jurnal-jurnal penelitian yang terkait dengan penelitian yang akan dilakukan. Tahapan telaah literatur ini menjadi tahapan yang paling awal adalah proses penelitian yang akan dilakukan.

2. Analisis Kebutuhan

Pada tahap ini, dilakukan analisis terhadap metode, data dan proses apa saja yang akan digunakan pada penelitian ini. Proses analisis dimulai dari teori yang akan digunakan dan menyusun struktur proses pada program yang akan dibuat. Pada tahap ini juga akan menganalisis bentuk keluaran apa saja yang akan dihasilkan oleh program.

3. Pengerjaan Program

Pada tahap ini dilakukan proses implementasi dari rancangan yang sudah dilakukan pada tahap-tahap sebelumnya. Tahap pertama yang akan dilakukan adalah *Web Scraping*. Tahap membuat program web scrapper untuk mengambil artikel berita yang dibutukan. Program disesuaikan dengan parameter, URL dan apikey yang sebelumnya telah diberikan hak akses kepada penulis oleh pihak yang telah terautorisasi. Dalam penelitian ini, *dataset* yang diteliti berasal dari artikel berita kompas.com dengan jumlah berita 10 ribu dokumen, yang kemudian data tersebut disimpan dalam format CSV untuk tahap pemrosesan berikutnya. Tahap selanjutnya yang dilakukan adalah *pre-processing* (*data cleaning*), meliputi *case folding, filtering, stemming, term weighting, standardization*. Data yang diambil dari tahap sebelumnya adalah data mentah memiliki *noise* atau kata-kata yang tidak dibutuhkan. Pada tahap ini data akan menjalani serangkaian proses pembersihan dan pembentukan sehingga data menjadi siap digunakan dan lebih efektif pada proses berikutnya.

4. Uji Coba dan Evaluasi

Data yang telah disiapkan sebelumnya akan diuji oleh Algoritma *Machine Learning* K-Means dan Gaussian Mixture Model, yang kemudian hasil dari pengujian 2 algoritma akan dievaluasi dengan Metode *Sillhouete Coefficient*.

5. Konsultasi dan Penulisan

Penulisan laporan dilakukan pada tahapan ini dengan tujuan mendokumentasikan segala bentuk proses penelitian serta menyimpulkan hasil akhir yang didapat dari pengerjaan tugas akhir ini.

3.2 Perancangan

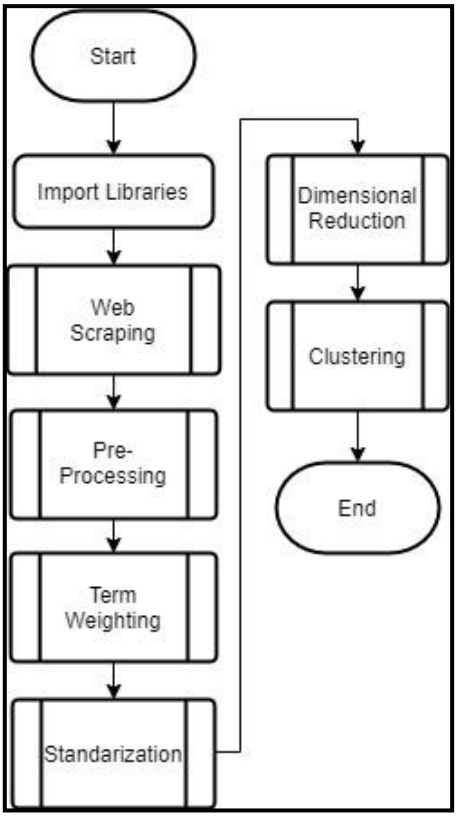
Aplikasi dirancang dengan menggunakan flowchart untuk menunjukkan alur program. Flowchart dari aplikasi ini adalah sebagai berikut.

3.2.1 Flowchart

Alur dari program dapat digambarkan dalam *flowchart* seperti yang terlihat pada Gambar 3.1 hingga Gambar 3.15. *Flowchart* dibagi menjadi empat belas yaitu *Flowchart utama*, *Flowchart Web Scrapping*, *Flowchart Preprocessing*, *Flowchart Merge Column*, *Flowchart Filter words*, *Flowchart Stopwords Removal*, *Flowchart Stemming*, *Flowchart OnlyAlpha*, *Flowchart wordLength*, *Flowchart OneSpace*, *Flowchart Term Weighting*, *Flowchart Standardization*, *Flowchart Dimensional Reduction*, *Flowchart Clustering*, *Flowchart Data Visualization*.

A. Flowchart Umum

Gambar 3.1 menunjukan *flowchart* secara umum yang digunakan dari awal pengambilan data hingga mengevaluasi hasil clustering. Proses dapat dibagi menjadi beberapa tahap yaitu pertama, persiapan program dimulai dengan mengimpor *library-library* yang diperlukan. Berikutnya adalah proses pengambilan dan pembersihan data (*gather and cleaning data*), *pre-processing*, *term weighting*, *standardization*, *dimensional reduction*, dan *clustering*.



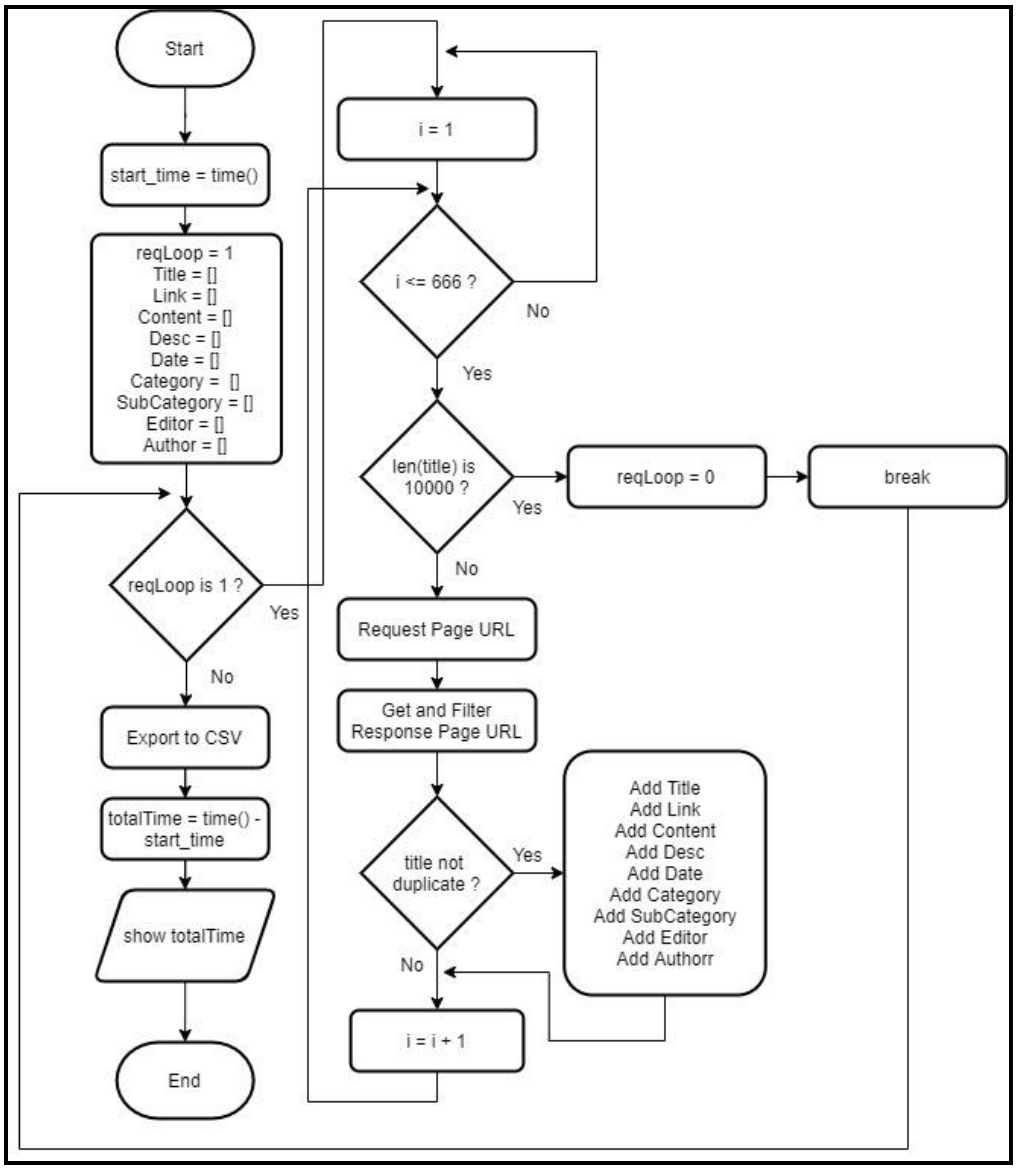
Gambar 3.1 Flowchart Utama

B. Flowchart Web Scrapping

Gambar 3.2 menunjukan *flowchart* proses pengambilan dan pembersihan data (*gather and cleaning data*) guna mendapatkan *raw* dataset. Proses dimulai dengan menjalankan fungsi *time* untuk mendapatkan waktu awal eksekusi. Selanjutnya, dilakukan inisialisasi variabel reqLoop dan sejumlah array yaitu *title*, *link*, *content*, *desc*, *date*, *category*, *subcategory*, *editor*, *author*. Proses dilanjutkan dengan iterasi yang terus dilakukan selama kondisi reqLoop sama dengan 1. Berikutnya, Jika kondisi reqLoop sama dengan 1, maka proses dilanjutkan dengan iterasi sebanyak 666 kali untuk mengambil keseluruhan data berita yang berjumlah 15 dari *request*

page tiap URLnya. Dalam setiap iterasinya, dilakukan pengecekan terhadap kondisi reqLoop, jika kondisi reqloop sama dengan 0, maka akan dijalankan fungsi break untuk keluar dari iterasi saat ini.

Hasil dari *request page* URL menghasilkan *response* yang kemudian akan dilakukan iterasi untuk disimpan kedalam array satu per satu. Dalam setiap iterasinya, dilakukan filter pada jumlah title yang telah tersimpan saat ini, jika jumlah title telah mencapai 10 ribu, maka variabel reqLoop akan diubah menjadi 0 dan menjalankan fungsi break untuk keluar dari iterasi saat ini. Jika jumlah title belum mencapai 10 ribu, maka data berita akan dimasukan ke dalam array *title*, *link*, *content*, *desc*, *date*, *category*, *subcategory*, *editor*, *author* dilanjutkan dengan proses iterasi berikutnya jika *title* terindentifikasi tidak duplikat, jika *title* teridentifikasi duplikat maka akan dilanjutkan dengan proses iterasi berikutnya tanpa pemasukan data berita ke dalam array. Jika kondisi reqLoop sama dengan 0, maka dataset yang telah dikumpulkan kemudian di export ke dalam bentuk CSV. Selanjutnya, dijalankan fungsi *time* untuk mendapatkan waktu selesai eksekusi dan menampilkan waktu total eksekusi dengan mengurangi waktu selesai eksekusi dengan start_time (waktu awal eksekusi).

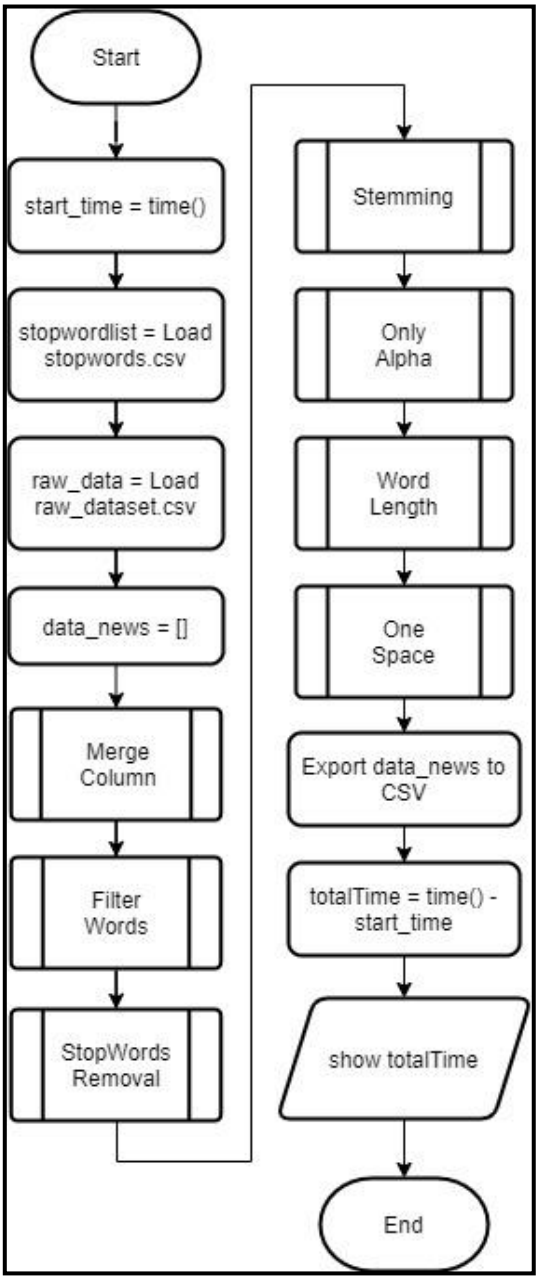


Gambar 3.2 Flowchart Web Scrapping

C. Flowchart Preprocessing

Gambar 3.3 menunjukan *flowchart preprocessing* yang digunakan untuk memproses *raw* dataset menjadi dataset yang siap diolah untuk proses selanjutnya. Proses dimulai dengan menjalankan fungsi *time* untuk mendapatkan waktu awal eksekusi. Selanjutnya, dilakukan proses *load raw* dataset yang berasal dari proses sebelumnya dan menginisialisasi array data_news. Proses berikutnya dibagi

menjadi beberapa tahap yaitu *merge column, filter words, stopwords removal, stemming, only alpha, word length* dan *one space*. Setelah dilakukan serangkaian proses *data cleaning*, dataset kemudian di export ke dalam bentuk CSV dan dijalankan fungsi *time* untuk mendapatkan waktu selesai eksekusi. Selanjutnya menampilkan waktu total eksekusi dengan mengurangi waktu selesai eksekusi dengan start_time (waktu awal eksekusi).

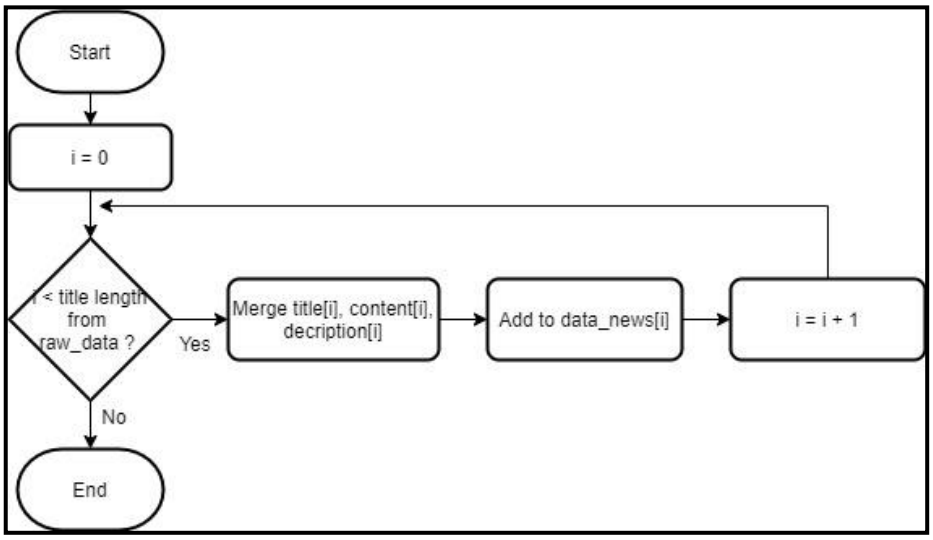


Gambar 3.3 Flowchart Preprocessing

D. Flowchart Merge Column

Gambar 3.4 menunjukan *flowchart merge column* yang digunakan untuk menggabungkan isi dari *title*, *content*, dan *description* berita guna memperkuat identitas suatu berita. Proses dimulai dengan iterasi sebanyak jumlah judul berita

dari *raw* dataset yang telah di *load*. Dalam setiap iterasinya, *title*, *content*, dan *description* berita yang telah diggabungkan kemudian ditambahkan ke dalam array data_news.



Gambar 3.4 Flowchart Merge Column

E. Flowchart Filter Words

Gambar 3.5 menunjukan *flowchart filter words* yang digunakan untuk membersihkan data berita dari format dan pola kata yang tidak diinginkan. Proses dimulai dengan menginisialisasi array temp dan iterasi sebanyak jumlah array data_news. Dalam setiap iterasinya, data berita dalam array data_news akan di-filter dari kata yang mengandung awalan huruf '<' dan akhiran huruf '>'. Data berita yang telah di-filter dan tidak mengandung pola kata '<…>' kemudian ditambahkan ke dalam array temp. Selanjutnya, isi array data_news akan di *replace* dengan isi array temp, dan isi array temp akan dikosongkan.

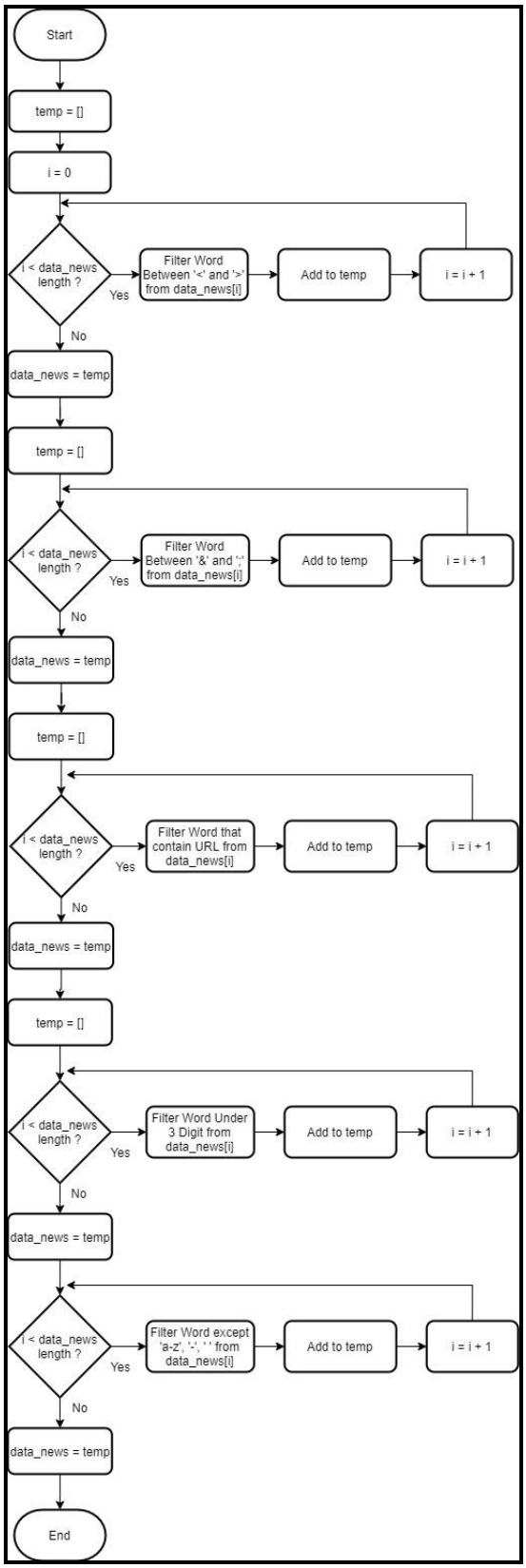
Proses selanjutnya, diawali iterasi kembali sebanyak jumlah array data_news. Dalam setiap iterasinya, data berita dalam array data_news akan di-filter dari kata yang mengandung awalan huruf '&' dan akhiran huruf ';'. Data berita yang telah di-filter dan tidak mengandung pola kata '&…;' kemudian ditambahkan ke dalam array temp. Selanjutnya, isi array data_news akan di *replace* dengan isi array temp, dan isi array temp akan dikosongkan.

Proses selanjutnya, diawali iterasi kembali sebanyak jumlah array data_news. Dalam setiap iterasinya, data berita dalam array data_news akan di-filter dari kata yang mengandung URL. Data berita yang telah di-filter dan tidak mengandung URL kemudian ditambahkan ke dalam array temp. Selanjutnya, isi array data_news akan di *replace* dengan isi array temp, dan isi array temp akan dikosongkan.

Proses selanjutnya, diawali iterasi kembali sebanyak jumlah array data_news. Dalam setiap iterasinya, data berita dalam array data_news akan di-filter dari kata yang mengandung karakter kurang dari 3. Data berita yang telah di-filter dan tidak

mengandung karakter kurang dari 3 kemudian ditambahkan ke dalam array temp. Selanjutnya, isi array data_news akan di *replace* dengan isi array temp, dan isi array temp akan dikosongkan.

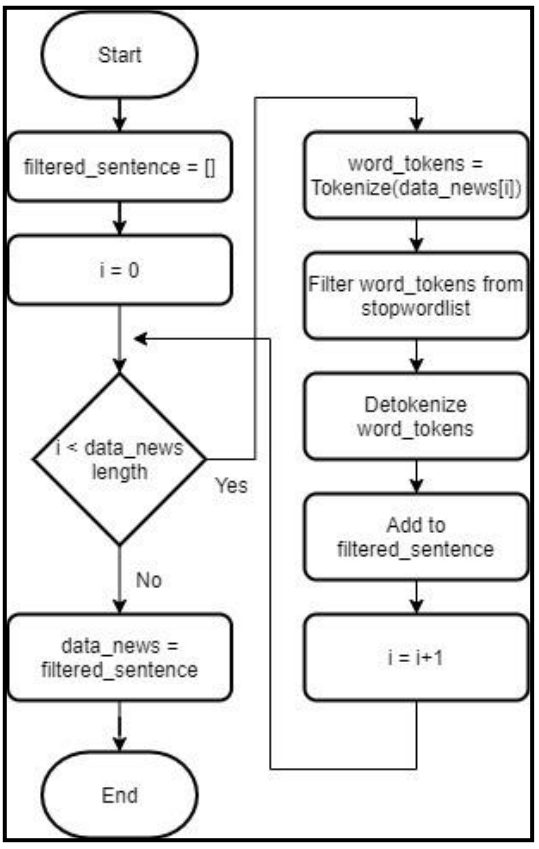
Proses selanjutnya, diawali iterasi kembali sebanyak jumlah array data_news. Dalam setiap iterasinya, data berita dalam array data_news akan di-filter dari kata yang mengandung selain karakter ‘a-z’, ‘-’, dan spasi. Data berita yang telah di-filter dan hanya mengandung karakter ‘a-z’, ‘-’, dan spasi kemudian ditambahkan ke dalam array temp. Selanjutnya, isi array data_news akan di *replace* dengan isi array temp, dan isi array temp akan dikosongkan.



Gambar 3.5 Flowchart Filter words

F. Flowchart Stopwords Removal

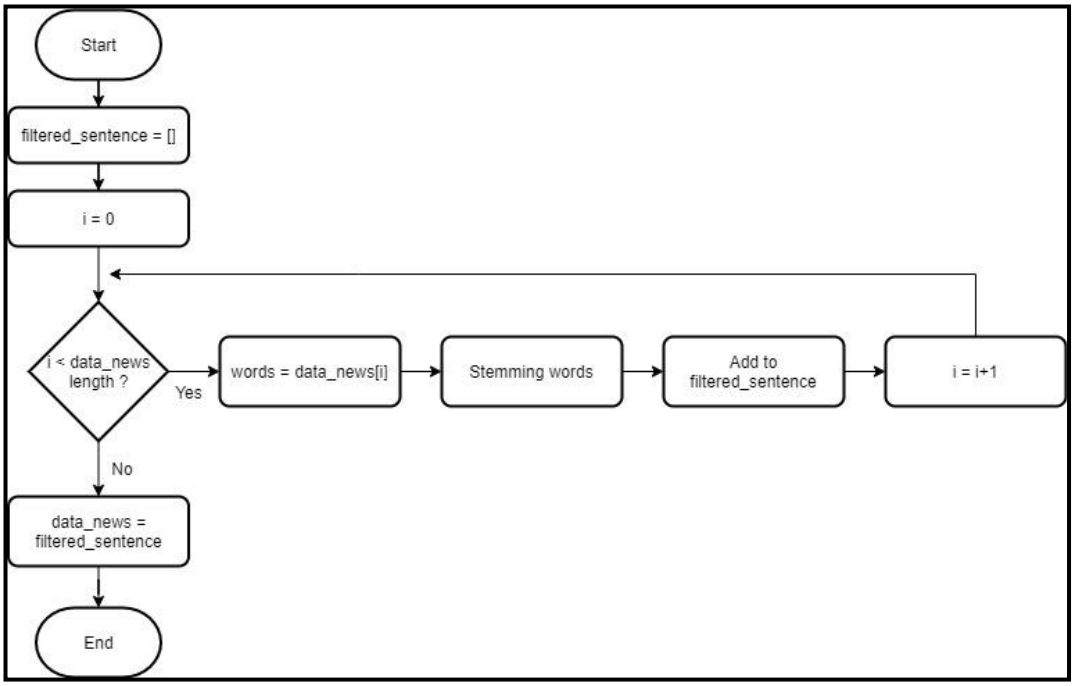
Gambar 3.6 menunjukan *flowchart stopwords removal* yang digunakan untuk membersihkan data berita dari *list* kata yang terdapat pada stopwords list. Proses dimulai dengan menginisialisasi array filtered_sentence dan iterasi sebanyak jumlah array data_news. Dalam setiap iterasinya, data berita dalam array data_news yang merupakan suatu *string* akan di pecah (*Tokenizing*) menjadi *list* kata dan di-filter berdasarkan stopwordlist. Selanjutnya, *list* kata yang telah di-filter akan digabungkan kembali (*Detokenizing*) dan ditambahkan ke dalam array filtered_sentence. Proses diakhiri dengan mereplace isi array data_news dengan array filtered_sentence.



Gambar 3.6 Flowchart Stopwords Removal

G. Flowchart Stemming

Gambar 3.7 menunjukan *flowchart stemming* yang digunakan untuk mengubah kata imbuhan dalam data berita menjadi bentuk kata dasarnya. Proses dimulai dengan menginisialisasi array filtered_sentence dan iterasi sebanyak jumlah array data_news. Dalam setiap iterasinya, data berita dalam array data_news akan di *stemming* dan kemudian ditambahkan ke dalam array filtered_sentence. Selanjutnya, isi array data_news akan di *replace* dengan array filtered_sentence.

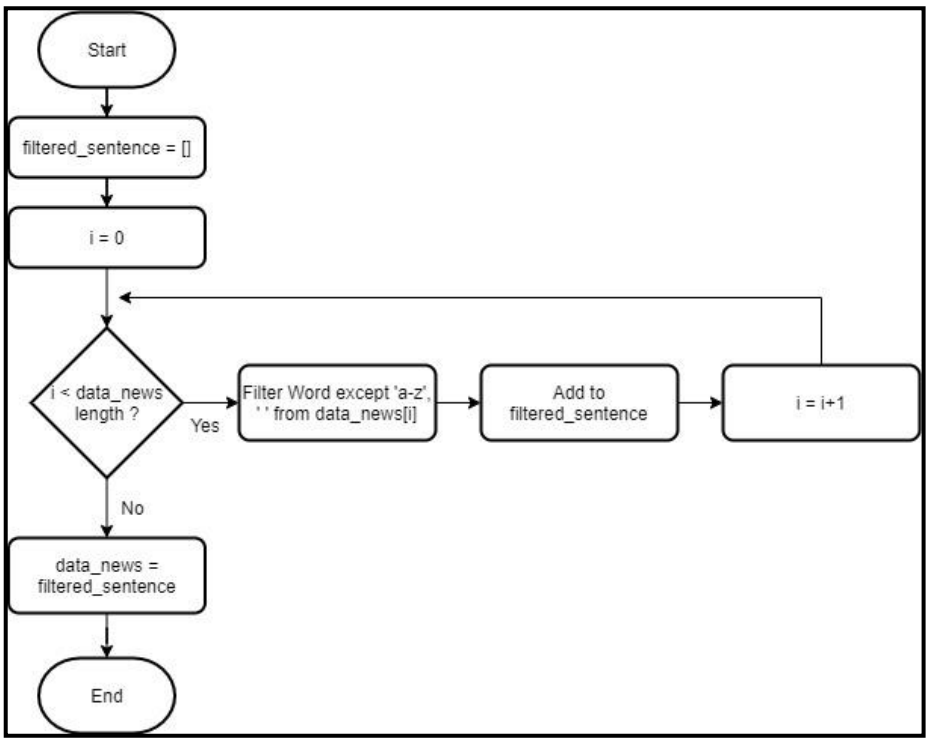


Gambar 3.7 Flowchart Stemming

H. Flowchart OnlyAlpha

Gambar 3.8 menunjukan *flowchart OnlyAlpha* yang digunakan untuk memfilter data berita agar hanya mengandung karakter huruf ‘a-z’ dan spasi. Proses dimulai dengan menginisialisasi array filtered_sentence dan iterasi sebanyak

jumlah array data_news. Dalam setiap iterasinya, data berita dalam array data_news akan di-filter agar hanya mengandung karakter huruf ‘a-z’ dan spasi, kemudian ditambahkan ke dalam array filtered_sentence. Selanjutnya, isi array data_news akan di *replace* dengan array filtered_sentence.

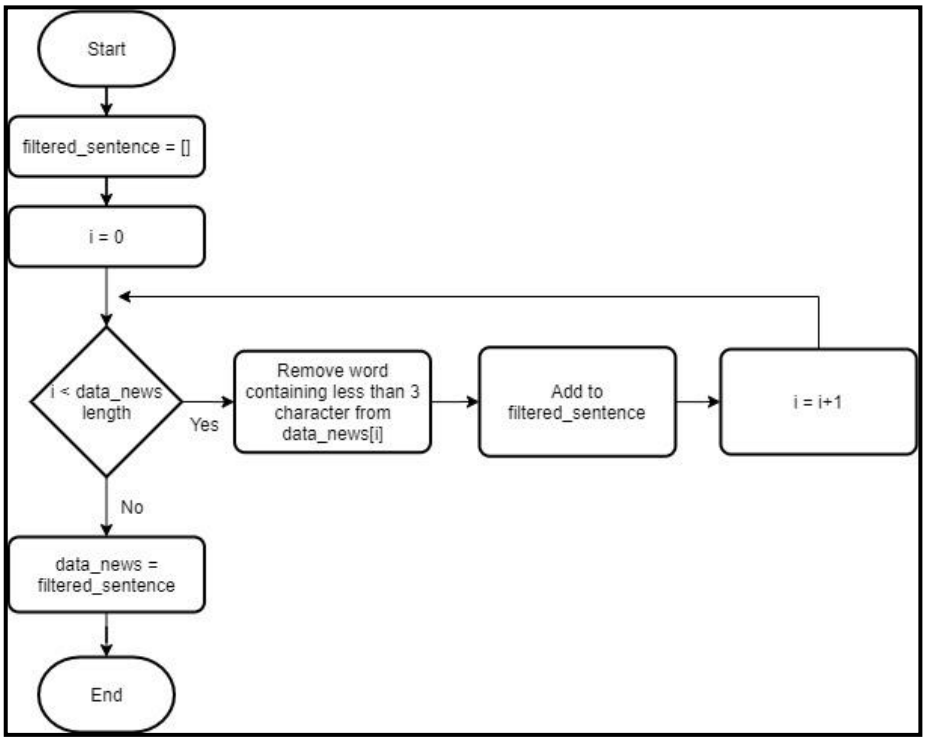


Gambar 3.8 Flowchart OnlyAlpha

I. Flowchart WordLength

Gambar 3.9 menunjukan *flowchart WordLength* yang digunakan untuk meremove kata yang mengandung kurang dari 3 karakter. Proses dimulai dengan menginisialisasi array filtered_sentence dan iterasi sebanyak jumlah array data_news. Dalam setiap iterasinya, data berita dalam array data_news akan di-filter agar hanya mengandung jumlah karakter diatas 2, kemudian ditambahkan ke

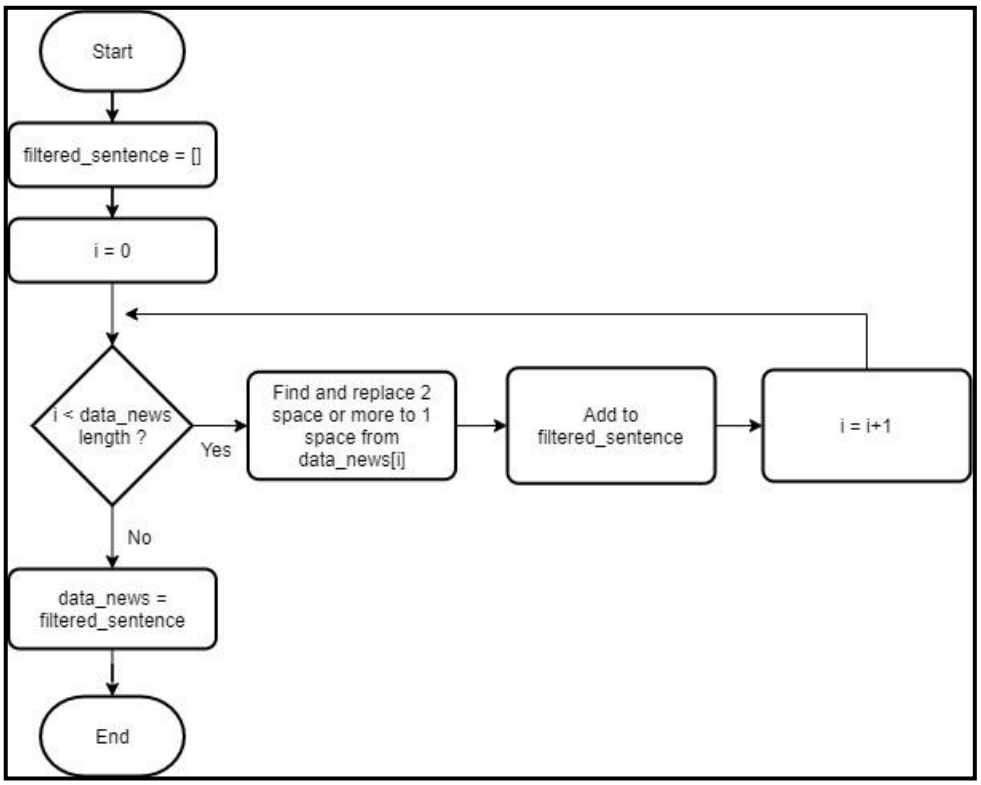
dalam array filtered_sentence. Selanjutnya, isi array data_news akan di *replace* dengan array filtered_sentence.



Gambar 3.9 Flowchart WordLength

J. Flowchart OneSpace

Gambar 3.10 menunjukan *flowchart OneSpace* yang digunakan untuk memfilter spasi antar kata agar hanya mengandung 1 spasi. Proses dimulai dengan menginisialisasi array filtered_sentence dan iterasi sebanyak jumlah array data_news. Dalam setiap iterasinya, data berita dalam array data_news yang memiliki lebih dari 1 spasi akan di *replace* dengan 1 spasi, kemudian ditambahkan ke dalam array filtered_sentence. Selanjutnya, isi array data_news akan di *replace* dengan array filtered_sentence.

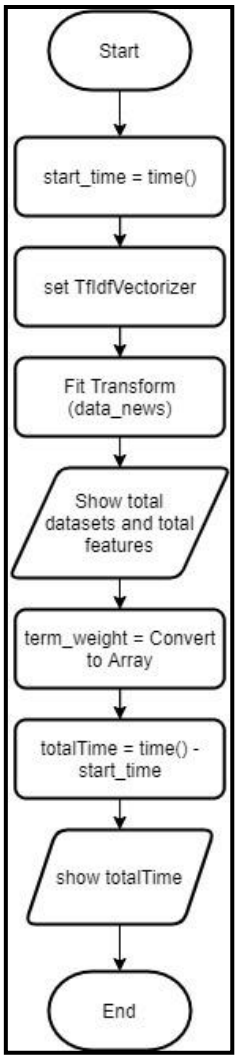


Gambar 3.10 Flowchart OneSpace

K. Flowchart Term Weighting

Gambar 3.11 menunjukan *Flowchart Term Weighting* yang digunakan untuk memberikan pembobotan nilai pada setiap data berita. Proses dimulai dengan menjalankan fungsi time untuk mendapatkan waktu awal eksekusi. Selanjutnya, set TfIdfVectorizer dan komputasi pembobotan dengan fungsi fit_transform pada array data_news. Proses dilanjutkan dengan menampilkan total dataset dan total features pada array data_news. Selanjutnya, array data_news yang telah di fit_transform dikonversikan ke dalam bentuk array dan disimpan dalam variabel term_weight. Proses diakhiri dengan dijalankan fungsi time untuk mendapatkan waktu selesai

eksekusi. Selanjutnya menampilkan waktu total eksekusi dengan mengurangi waktu selesai eksekusi dengan start_time (waktu awal eksekusi).

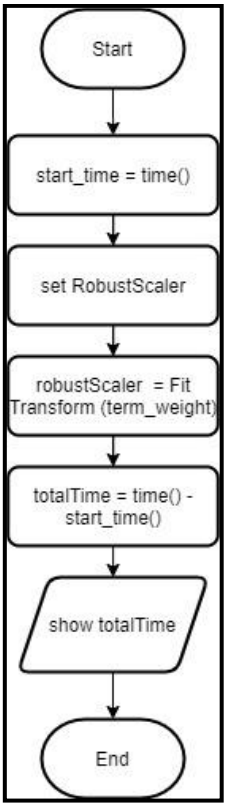


Gambar 3.11 Flowchart Term Weighting

L. Flowchart Standardization

Gambar 3.12 menunjukan *Flowchart Standardization* yang digunakan untuk menskalakan ulang nilai bobot untuk memastikan nilai *mean* menjadi 0 dan

standard deviation menjadi 1. Proses dimulai dengan menjalankan fungsi time untuk mendapatkan waktu awal eksekusi. Selanjutnya, set RobustScaler dan komputasi pembobotan ulang dengan fungsi fit_transform pada array term_weight. Proses diakhiri dengan dijalankan fungsi time untuk mendapatkan waktu selesai eksekusi. Selanjutnya menampilkan waktu total eksekusi dengan mengurangi waktu selesai eksekusi dengan start_time (waktu awal eksekusi).

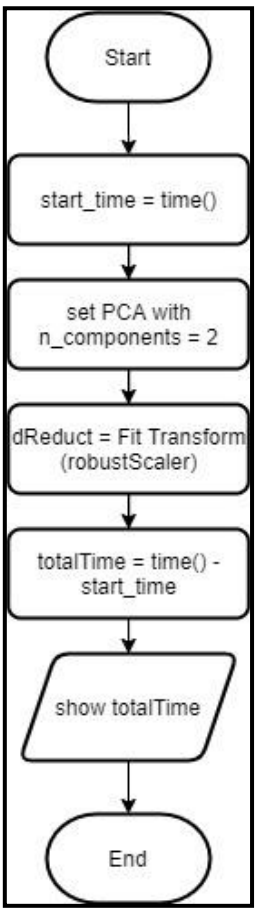


Gambar 3.12 Flowchart Standardization

M. Flowchart Dimensional Reduction

Gambar 3.13 menunjukan *Flowchart Dimensional Reduction* yang digunakan untuk mereduksi data dimensi tinggi menjadi 2 dimensi. Proses dimulai dengan

menjalankan fungsi time untuk mendapatkan waktu awal eksekusi. Selanjutnya, set n_component (jumlah dimensi) menjadi 2 dan komputasi reduksi dimensi dengan fungsi fit_transform pada array robustScaler. Proses diakhiri dengan menjalankan fungsi time untuk mendapatkan waktu selesai eksekusi. Selanjutnya menampilkan waktu total eksekusi dengan mengurangi waktu selesai eksekusi dengan start_time (waktu awal eksekusi).

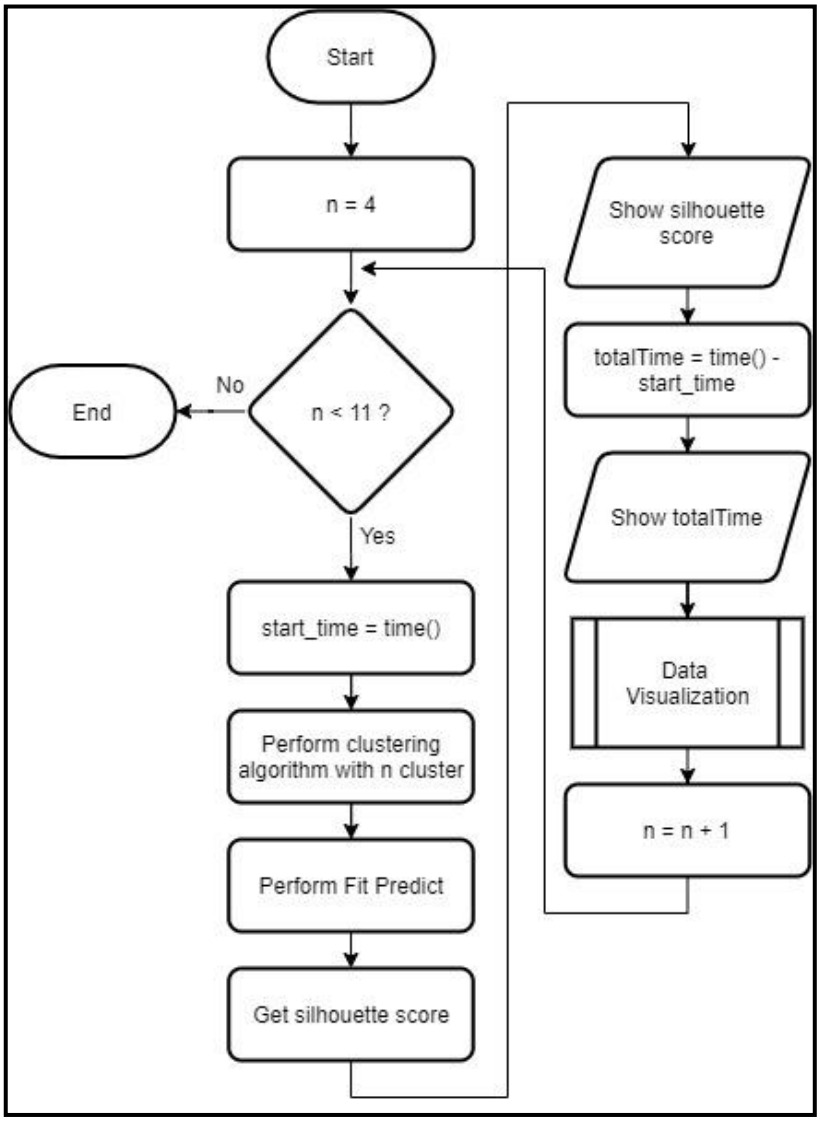


Gambar 3.13 Flowchart Dimensional Reduction

N. Flowchart Clustering

Gambar 3.14 menunjukan *Flowchart Clustering* yang digunakan untuk mengelompokan data berita menjadi beberapa *cluster* dan mengevaluasi hasil dari

clustering. Proses dimulai dengan melakukan set nilai variabel n (jumlah *cluster*) menjadi 4 dan menjalankan iterasi sebanyak 6 kali hingga n menjadi 10. Dalam setiap iterasinya, fungsi time dijalankan untuk mendapatkan waktu awal eksekusi dan menjalankan algoritma *clustering* dengan jumlah n *cluster*. Proses dilanjutkan dengan menghitung pusat cluster dan prediksi index cluster untuk setiap data berita (Fit Predict). Selanjutnya dilakukan proses perhitungan nilai silhouette score dan menampilkan hasil nilai silhouette score. Proses dilanjutkan dengan menjalankan fungsi time untuk mendapatkan waktu selesai eksekusi dan menampilkan waktu total eksekusi dengan mengurangi waktu selesai eksekusi dengan start_time (waktu awal eksekusi). Proses diakhiri dengan menampilkan hasil visualisasi *clustering* melalui *scatter plot*.

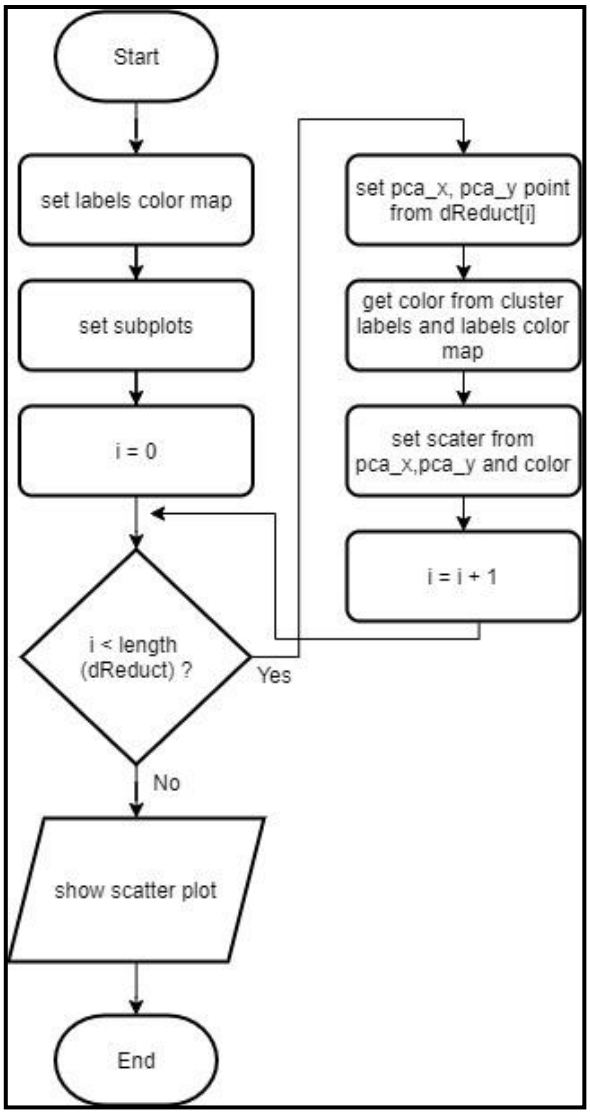


Gambar 3.14 Flowchart Clustering

O. Flowchart Data Visualization

Gambar 3.15 menunjukan *Flowchart Data Visualization* yang digunakan untuk menampilkan hasil visualisasi *clustering* melalui *scatter plot*. Proses dimulai dengan melakukan set nilai hexadecimal pada label_color_map dan set nilai subplots. Proses dilanjutkan dengan iterasi sebanyak jumlah array dReduct. Dalam setiap iterasinya, dilakukan set nilai pca_x dan pca_y dari nilai array dReduct dan

set color berdasarkan cluster labels dan nilai index pada labels color map. Selanjutnya, set nilai scatter plot dengan pca_x, pca_y dan color. Proses diakhiri dengan menampilkan hasil visualisasi cluster melalui scatter plot.



Gambar 3.15 Flowchart Data Visualization