BAB 3

METODOLOGI PENELITIAN

3.1 Metodologi Penelitian

Agar penelitian dapat berjalan secara sistematis, penelitian dilakukan dalam beberapa langkah secara bertahap. Setiap tahap penelitian dapat diuraikan menjadi beberapa poin sebagai berikut:

1) Studi Literatur

   Studi literatur menjadi tahap awal dari proses penelitian. Hal ini dilakukan dengan mencari, membaca, dan mempelajari beberapa sumber jurnal ilmiah atau karya tulis ilmiah lainnya. Tujuannya agar dapat meningkatkan pemahaman penulis terkait teori mengenai metode yang digunakan dalam penelitian ini.

2) Analisis Kebutuhan

   Proses ini dilakukan bersamaan dengan studi literatur. Hal ini dilakukan agar penulis dapat memahami hal-hal apa saja yang diperlukan untuk membentuk sistem terkait pernelitian berdasarkan informasi yang diperoleh selama studi literatur. Tahap ini termasuk juga dengan mengumpulkan *dataset*.

3) Perancangan dan Implementasi Sistem

Hal ini dilakukan dengan memahami dan menggambarkan alur logika program. Alur logika sistem digambarkan ke dalam bentuk *flowchart*. Setelah alur tersusun secara sistematis, implementasi dapat dilakukan. Dimulai dengan melakukan *preprocessing dataset* dan membentuk model *bagging* yang digabungkan dengan *Multinomial Naïve Bayes* dan *TF-IDF* sebagai metode ekstraksi fitur.

4) *Testing and Debugging*

Model yang dibuat kemudian dilatih dengan *dataset* yang sudah disediakan. Selain itu dilakukan penyesuaian *source code* untuk meminimalisir *runtime* dan apabila terjadi kesalahan.

5) Evaluasi

Berdasarkan pengujian yang dilakukan, dilakukan evaluasi model yang dihitung dengan metrik *f1-score*. Berdasarkan hasil tersebut dibuat grafik untuk membandingkan sekaligus melihat kinerja antar model yang dihasilkan yang menjadi tujuan utama dari penelitian ini.

6) Dokumentasi

Hasil yang didapatkan kemudian dituliskan ke dalam bentuk laporan. Dokumentasi dilakukan secara bertahap mulai dari pendahuluan hingga kesimpulan. Selama proses dokumentasi, dilakukan juga konsultasi dengan dosen pembimbing agar laporan dapat tersusun dengan baik dan benar.

3.2 Gambaran Umum Sistem

Sistem yang dibuat merupakan secara umum merupakan implementasi penggabungan dari teknik *bagging* dan *Multinomial Naïve Bayes*. Namun, algoritma tambahan lainnya seperti *TF-IDF* dan *F1-Score* juga diperlukan sebagai metode ekstraksi fitur dan mengevaluasi model. Langkah awal diawali dengan membaca *dataset* terlebih dahulu.

Dataset yang digunakan merupakan *IMDB Spoiler Dataset* yang didapatkan dari situs https://www.kaggle.com/rmisra/imdb-spoiler-dataset . *Dataset* ini sudah pernah digunakan dalam penulisan karya ilmiah lainnya. Sreejita Biswas dengan karya ilmiahnya yang berjudul "Spoiler Detection with Applied Machine Learning" menggunakan *dataset* yang diambil dari situs yang sama, yang berisi dua bagian yaitu *movie details* dan *user reviews* (2020). Pada penelitian ini, *dataset* yang digunakan merupakan *user reviews dataset*. Gambar 3.1 merupakan gambaran sekilas isi dari *user reviews dataset*.

	review_date	movie_id	user_id	is_spoiler	review_text	rating	review_summary
0	10 February 2006	tt0111161	ur1898687	True	In its Oscar year, Shawshank Redemption (writt...	10	A classic piece of unforgettable film-making.
1	6 September 2000	tt0111161	ur0842118	True	The Shawshank Redemption is without a doubt on...	10	Simply amazing. The best film of the 90's.
2	3 August 2001	tt0111161	ur1285640	True	I believe that this film is the best story eve...	8	The best story ever told on film
3	1 September 2002	tt0111161	ur1003471	True	**Yes, there are SPOILERS here**This film has ...	10	Busy dying or busy living?
4	20 May 2004	tt0111161	ur0226855	True	At the heart of this extraordinary movie is a ...	8	Great story, wondrously told and acted

Gambar 3.1 Gambar tabel *user reviews dataset*

User reviews dataset memuat setidaknya 400 ribu data dengan nilai *True* pada label *spoiler* dan 150 ribu data dengan nilai *False* sehingga dapat ditotal lebih dari 550 ribu data di dalamnya. Dilakukan *resampling* secara acak untuk masing-masing data sebanyak 30 ribu sehingga total data menjadi 60 ribu. Tujuannya untuk mengurangi waktu eksekusi *program* saat dilakukan *hyperparameter tuning* dan juga *cross validation* dikarenakan keterbatasan spesifikasi *hardware*.

Setelah *dataset* dibaca oleh sistem, dibuat *pipeline* untuk membentuk model *machine learning* sekaligus algoritma lainnya yang digunakan untuk *preprocessing data* dan ekstraksi fitur dengan *TF-IDF*. Setelah membentuk *pipeline* yang akan digunakan sebagai model, ditentukan beberapa parameter yang akan digunakan sebagai pertimbangan untuk mencari model dengan akurasi terbaik. Setelah model dan beberapa parameter pembentuk model ditentukan, dilakukan *hyperparameter tuning* untuk menentukan parameter terbaik yang akan menghasilkan model dengan tingkat akurasi yang lebih baik.

Hyperparameter tuning dapat dilakukan dengan menguji model secara berulang-ulang dengan parameter yang berbeda-beda. Langkah tersebut dapat disederhanakan dengan *GridSearchCV library* yang tidak hanya melakukan *hyperparameter tuning* namun juga melakukan *Cross Validation* untuk mendapatkan model dengan hasil yang lebih baik lagi dan juga melakukan evaluasi model berdasarkan metrik *F1*.

Setelah selesai dieksekusi, didapatkan beberapa akurasi model dengan beberapa parameter yang berbeda-beda. Dilakukan *plotting* untuk melihat perbedaan antar masing-masing model secara jelas. Berdasarkan hasil *plotting*, dapat dibuat kesimpulan terkait model yang hasilkan dengan algoritma *Bagging* dan *Multinomial Naïve Bayes*.

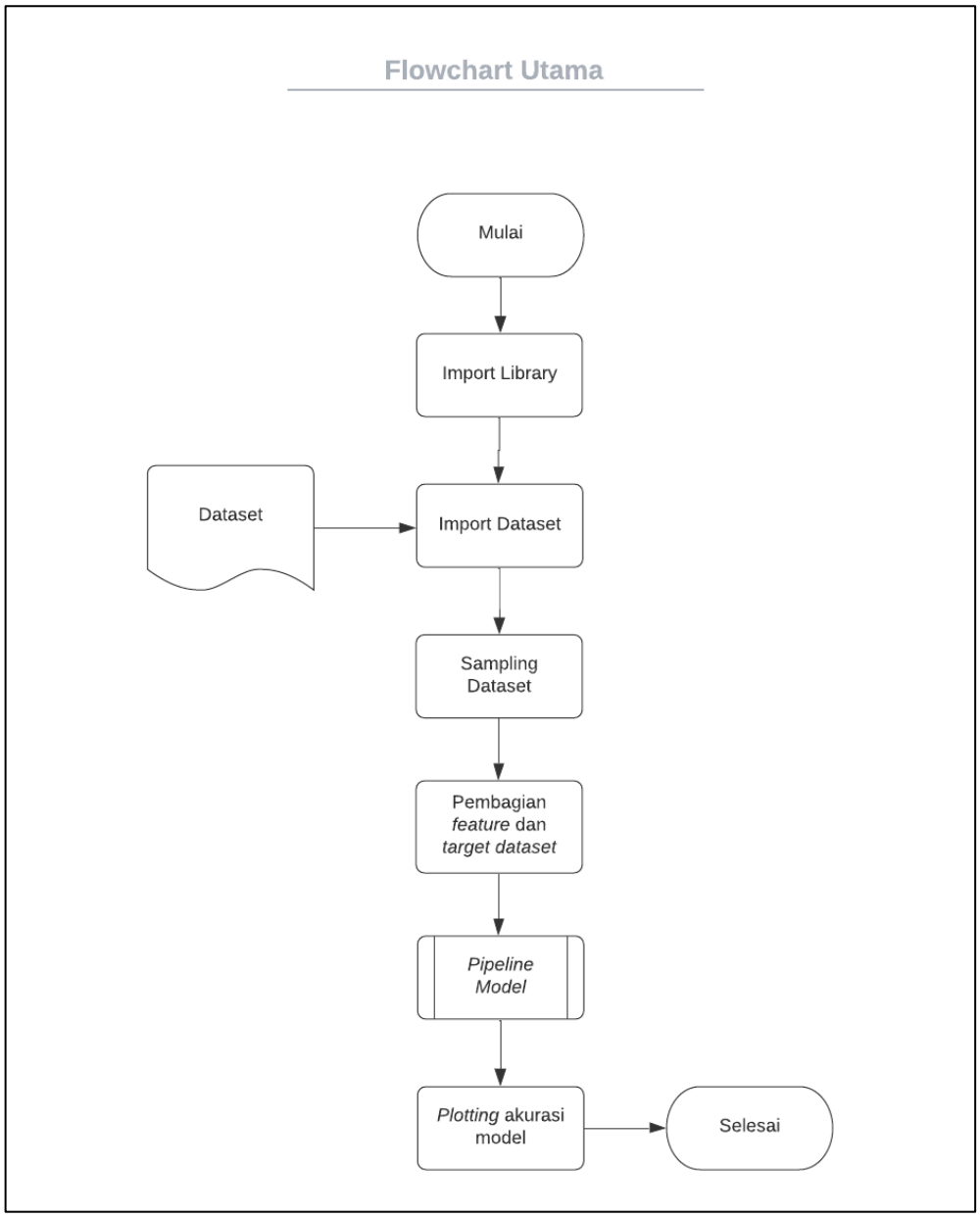
3.3 Flowchart

Proses berjalannya program dapat dijelaskan dalam bentuk diagram alur yang terdiri dari beberapa bagian: *flowchart* utama, *flowchart pipeline model*, *flowchart preprocessing, flowchart TF-IDF, flowchart Bagging with Multinomial Naïve Bayes*.

3.3.1 Flowchart Utama

Gambar 3.2 merupakan gambar diagram alur utama yang menjelaskan alur kerja program secara umum. Langkah pertama diawali dengan melakukan *import library* yang diperlukan untuk menjalankan program. Setelah itu, dilakukan *import dataset* dan *resampling dataset* ke dalam jumlah yang lebih kecil. *Dataset* kemudian dibagi menjadi data uji dan data latih.

Setelah itu, dilanjutkan dengan membuat model *pipeline* yang akan memproses data untuk tahap *preprocessing* sekaligus menjadi model *machine learning* yang akan diuji. Setelah model terbentuk, dilakukan *training* dan *testing* pada model secara berulang dengan parameter yang berbeda, atau disebut juga dengan *hyperparameter tuning* yang dibantu dengan *grid search cross validation*. *Plotting* dilakukan untuk melihat perbandingan antar akurasi model yang dihasilkan.

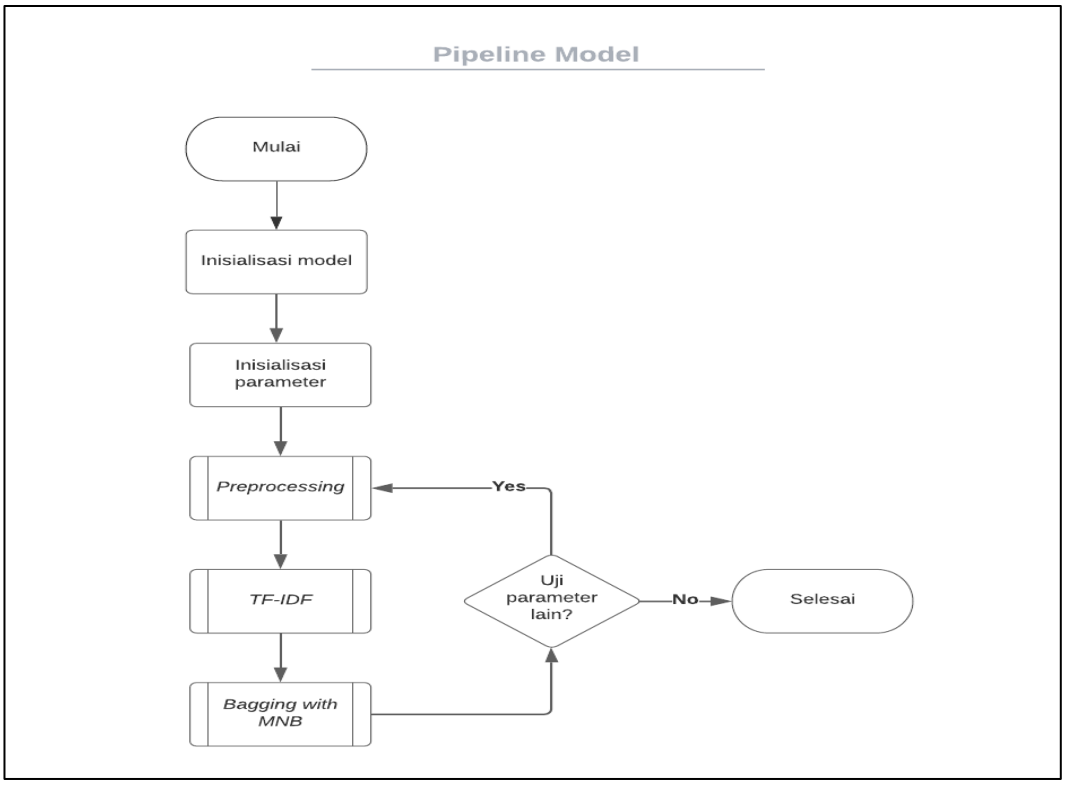


Gambar 3.2 *Flowchart* utama

3.3.2 Flowchart Pipeline Model

Gambar 3.3 merupakan gambar *flowchart pipeline model*. Dimulai dengan inisialisasi model dan parameter yang akan digunakan untuk *parameter tuning*. Setelah model dan parameter ditentukan, dilakukan *text preprocessing* untuk mengolah *dataset*. Data yang sudah melalui *text preprocessing* selanjutnya dapat digunakan untuk ekstraksi fitur dengan *TF-IDF*.

Setelah melalui *text preprocessing* dan ekstraksi fitur dengan *TF-IDF*, pengujian model *bagging* dengan *Multinomial Naïve Bayes* dapat dilakukan. Dengan bantuan *library GridsearchCV*, pembentukkan model baru dengan parameter yang berbeda dilakukan secara berulang-ulang untuk menemukan model dengan akurasi terbaik. Pengulangan ini dilakukan hingga seluruh kombinasi parameter telah diuji.

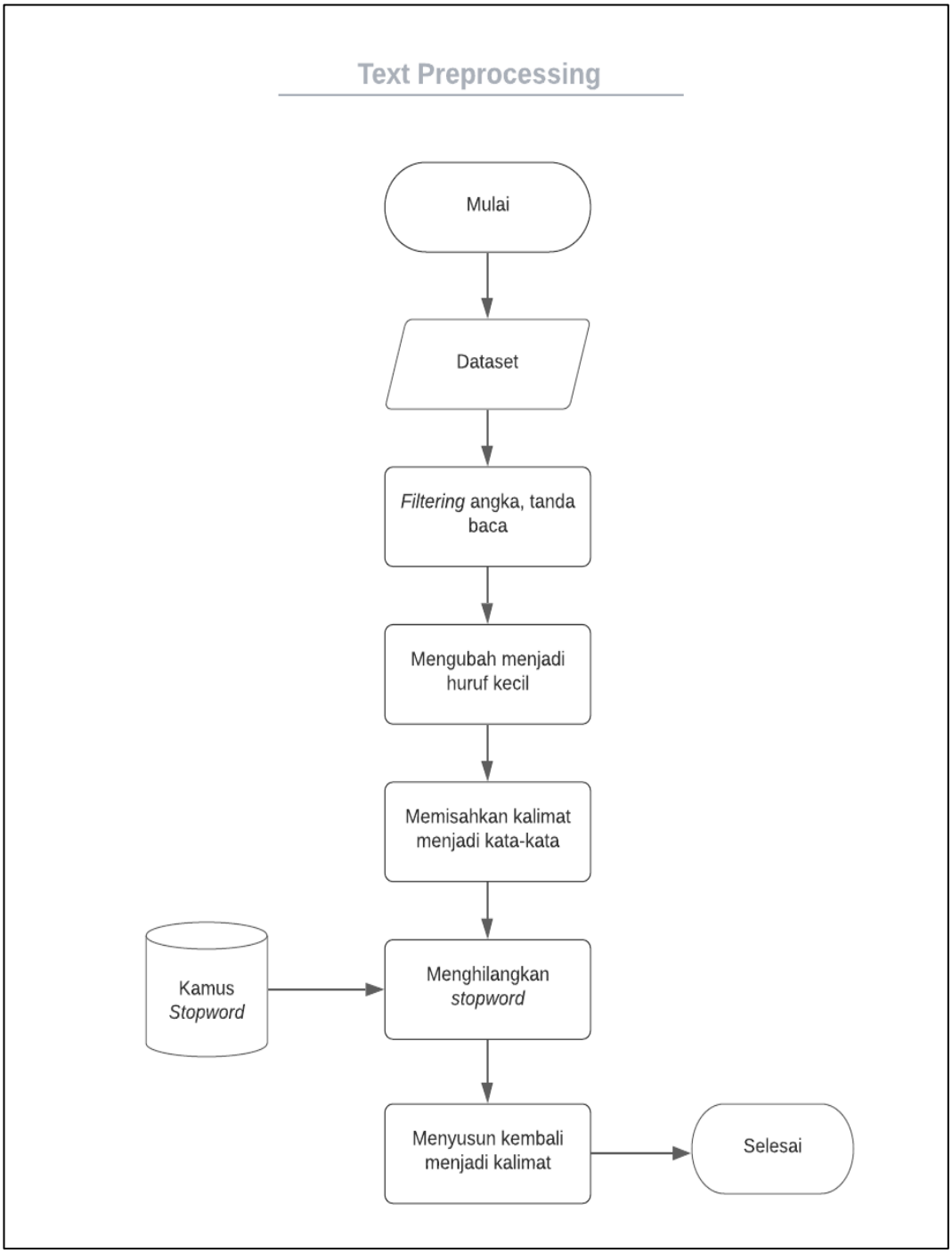


Gambar 3.3 *Flowchart Pipeline Model*

3.3.3 Flowchart Preprocessing

Gambar 3.4 merupakan gambar diagram alur untuk proses *text preprocessing* yang terdapat pada gambar 3.3. Dimulai dengan menghilangkan angka dan tanda baca yang tidak diperlukan. Setelah itu dilakukan *case folding* atau pengubahan huruf menjadi huruf kecil atau *lowercase*. Kalimat yang sudah diubah menjadi *lowercase,* diubah ke dalam kata-kata yang menyusun kalimat tersebut atau disebut juga *token*. Tahap ini disebut juga dengan *tokenizing*.

Token tersebut kemudian dibandingkan dengan kamus kata *stopwords* untuk melihat apakah *token* tersebut termasuk dalam kata *stopword* atau tidak. Jika termasuk dalam kata-kata *stopword* maka, kata atau *token* tersebut akan dihilangkan. Proses ini disebut juga dengan *stopwords removal*. Semua tahap *text preprocessing* tersebut, dirangkum dalam *library* ekstraksi fitur *TF-IDF* yang disediakan oleh *scikit-learn*.

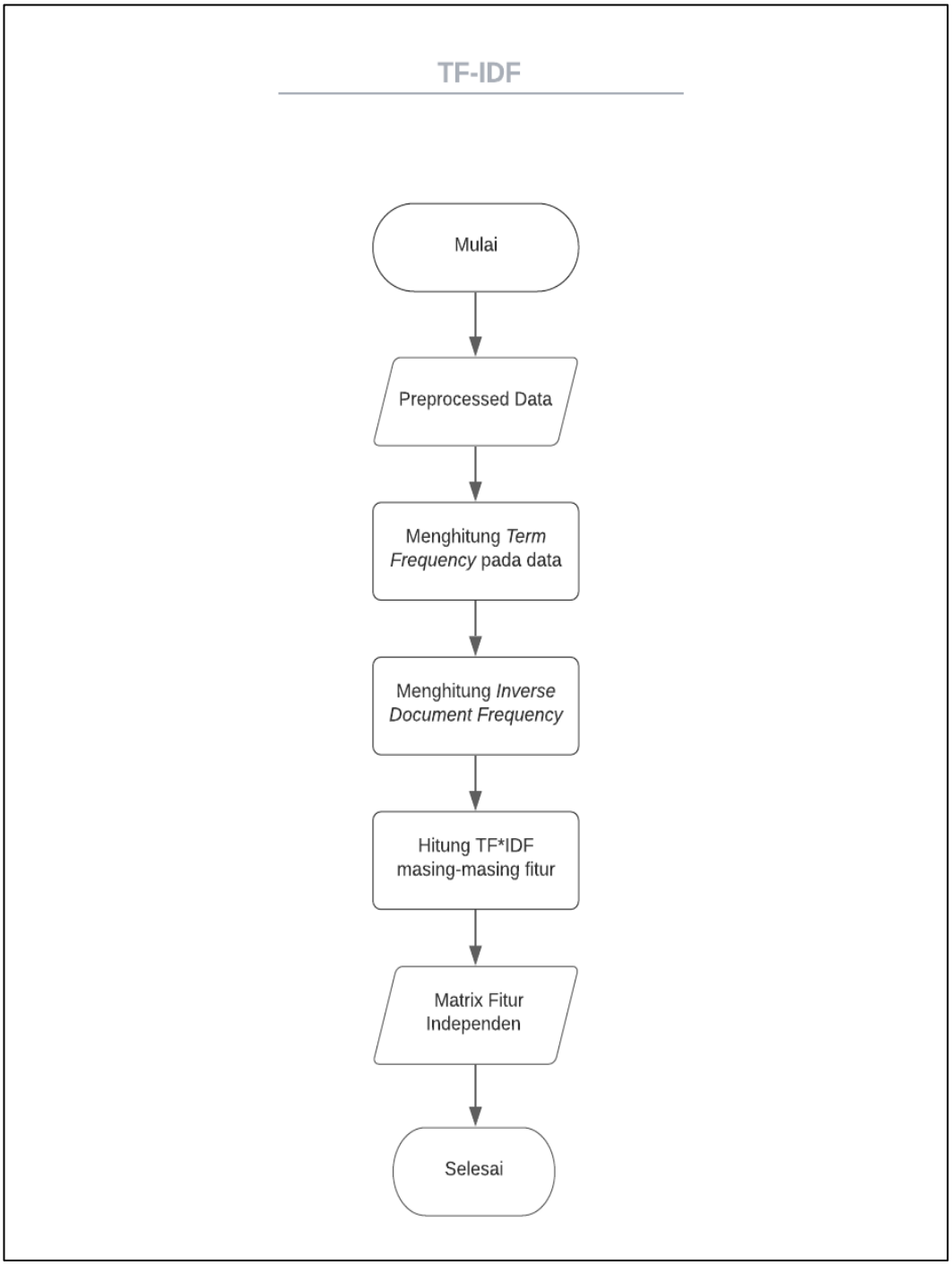


Gambar 3.4 *Flowchart preprocessing*

3.3.4 Flowchart TF-IDF

Gambar 3.5 merupakan gambar diagram alur untuk proses *TF-IDF* yang terdapat pada gambar 3.3. Data yang akan dihitung merupakan data yang sudah melalui tahap *preprocessing*. Langkah pertama diawali dengan menghitung *Term Frequency* kata dalam suatu dokumen yang terdapat dalam suatu *corpus*.

Setelah itu, dilakukan perhitungan untuk *IDF* atau *Inverse Document Frequency* sesuai dengan *IDF*. Setelah didapatkan masing-masing nilai *TF* dan *IDF* untuk masing-masing fitur dalam suatu kalimat, dapat dihitung nilai semantik atau nilai *TF-IDF* untuk suatu fitur dengan mengalikan nilai *TF* dan *IDF* sesuai dengan rumus *TF-IDF* yang sudah ditetapkan. Berdasarkan perhitungan tersebut, didapatkan matrix yang berisi nilai semantik untuk masing-masing fitur terhadap masing-masing dokumen yang ada pada suatu *corpus*.

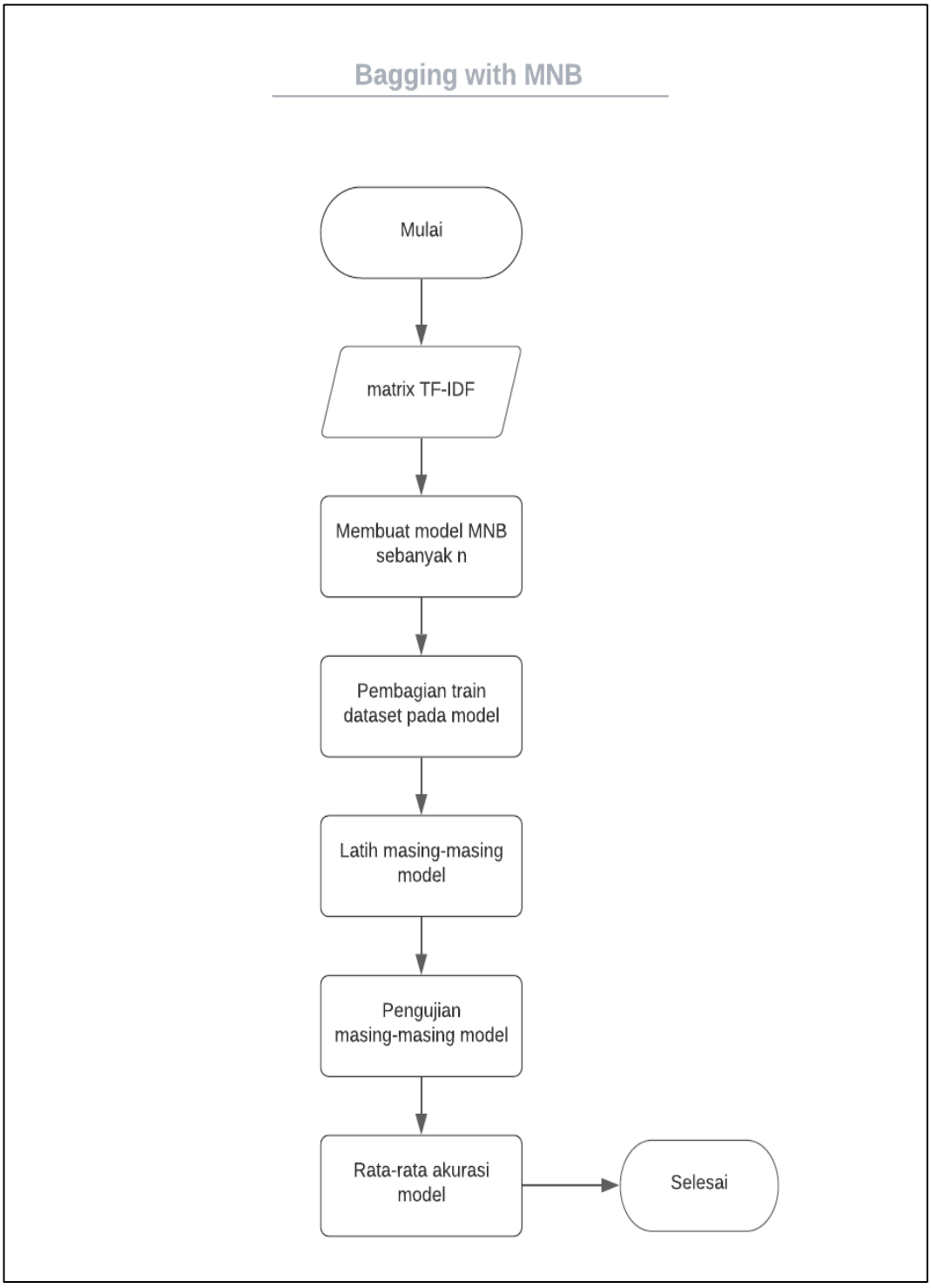


Gambar 3.5 *Flowchart TF-IDF*

3.3.5 Flowchart Bagging with Multinomial Naïve Bayes

Gambar 3.6 merupakan gambar diagram alur dari cara kerja model *Bagging* dengan *Multinomial Naïve Bayes*. Data hasil ekstraksi fitur digunakan sebagai data *training* untuk model yang akan dibentuk. Metode *bagging* membuat model *Multinomial Naïve Bayes* sejumlah *n* dimana *n* merupakan parameter yang kita tentukan. Nilai *n* akan diubah-ubah saat melakukan *parameter tuning*.

Setelah itu, masing-masing model mendapatkan masing-masing data latih yang dipilih dengan metode *random with replacement*. Artinya, data latih yang diambil dipilih secara random dan dapat dipilih kembali lebih dari satu kali dalam satu model yang sama. Selanjutnya, dilakukan *training* untuk masing-masing model yang dibentuk. Setelah melalui proses *training*, masing-masing model diuji menggunakan data uji yang sama untuk dilihat hasilnya pada masing-masing model. Hasil akhir didapatkan dengan melakukan perhitungan rata-rata hasil pengujian dari masing-masing model. Berdasarkan hasil akhir pengujian, dapat dilakukan evaluasi model dengan *F1-Score*.



Gambar 3.6 *Flowchart Bagging with Multinomial Naive Bayes*