BAB 3
METODOLOGI PENELITIAN

3.1. Metodologi Penelitian

Dalam pelaksanaan penelitian implementasi algoritma *multinomial naïve bayes* unuk analisis sentimen terhadap vaksinasi COVID-19 pada media sosial *Twitter* akan menerapkan metodologi penelitian sebagai berikut:

3.1.1. Studi Literatur

Studi literatur dilakukan sebagai tahap awal untuk mempelajari dan mengenal lebih dalam mengenai teori-teori terkait penelitian yaitu Vaksin COVID-19, *Labelling*, Analisis Sentimen, *Text Pre-processing*, *Term Frequency - Inverse Document Frequency* (TF-IDF), algoritma *Multinomial Naïve Bayes*, *Confusion Matrix*, dan *K-Fold Cross Validation*.

3.1.2. Pengumpulan Data

Tahap ini akan dilakukan pengumpulan data berupa *Tweet* dari pengguna *Twitter* Indonesia mengenai opininya terhadap vaksinasi COVID-19. Pengumpulan data dilakukan dengan menggunakan bantuan *library Twint* yang merupakan *tool* untuk mengambil *tweet* dari *twitter* secara mudah, tanpa melakukan *sign in*. Penelitian ini tidak dapat menggunakan *Twitter API (Application Programming Interface)* dikarenakan *Twitter API* hanya menyediakan data *tweet* yang paling terbaru dalam tujuh hari terakhir. Sedangkan yang dibutuhkan adalah data *tweet* dimulai dari 13 Januari 2021 hingga akhir tahun 2021. Oleh karena itu, penelitian ini menggunakan *library Twint*. Terdapat kelemahan dari penggunaan *library Twint*, yaitu jumlah *tweet* yang ditarik tidak dapat berjumlah sesuai dengan seperti menggunakan *Twitter API*. Hal ini dikarenakan *library Twint* memiliki limitasi terhadap jumlah *tweet*, yang hanya berjumlah sekitar 3200 *tweet* paling terakhir.

3.1.3. Perancangan Sistem

Pada tahap selanjutnya setelah mengumpulkan data, akan dilakukan perancangan sistem dimana perancangan sistem akan digambarkan terlebih

dahulu dalam bentuk diagram alur (*flowchart*) untuk membantu mempermudah pembacaan cara kerja pengimplementasian algoritma *Multinomial Naïve Bayes Classifier* dalam analisis sentimen. Hal ini juga membantu penulis agar pengerjaan penelitian dapat sesuai acuan yang ada pada diagram alur.

3.1.4. Pemrograman Sistem

Pemrograman sistem dilakukan sesuai dengan perancangan sistem yang telah digambarkan sebelumnya pada diagram alur. Tahap ini penulis akan melakukan *coding* dalam menganalisis data, pemrosesan data teks, dan menerapkan TF-IDF *Vectorizer* serta algoritma *Multinomial Naïve Bayes Classifier*.

3.1.5. Pengujian dan Evaluasi

Setelah model telah diimplementasikan, selanjutnya akan dilakukan pengujian sebagai bentuk pengecekan apakah model yang diterapkan untuk analisis sentimen ini berjalan dengan baik atau tidak. Pengujian dan evaluasi dilakukan berulang dengan mencoba menggunakan beberapa skenario yang berbeda untuk mendapatkan hasil yang terbaik. Hasil evaluasi dan validasi akan dihitung dengan menggunakan *confusion matrix*. Selain itu, validasi akan dilakukan dengan menggunakan bantuan *K-Fold Cross Validation* dengan jenis *stratified* agar setiap kelas terbagi secara rata untuk setiap *fold*-nya. Metrik evaluasi utama yang digunakan untuk pembacaan hasil evaluasi dan validasi dari berbagai pengujian adalah *accuracy* dan *f1-score* yang kemudian disimpulkan manakah pengujian yang dapat memberikan hasil klasifikasi terbaik.

3.1.6. Penulisan Laporan

Tahap terakhir pada metodologi penelitian ini adalah penulisan laporan. Penulisan laporan akan berisikan hasil penelitian yang telah dilakukan. Proses penulisan laporan akan dibuat mulai dari pendahuluan hingga kesimpulan dan saran mengenai penelitian yang telah dilakukan.

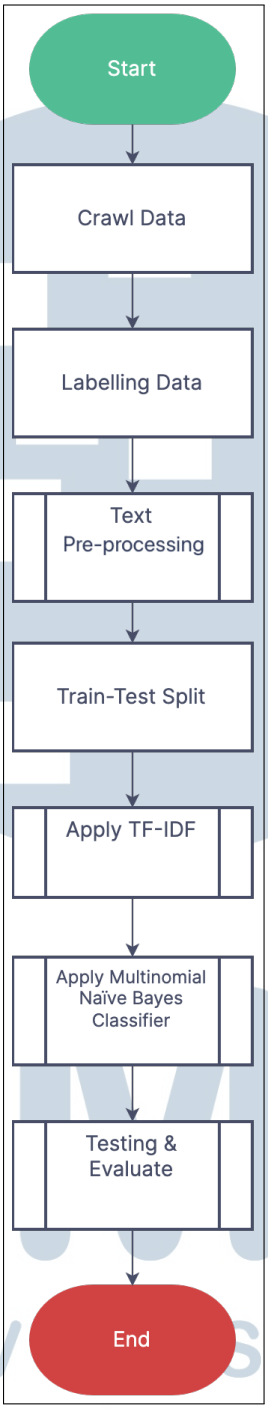
3.2. Perancangan Sistem

Pada bagian ini, akan dijabarkan masing-masing diagram alur dari sistem yang dibuat. Diagram-diagram alur tersebut antara lain adalah *flowchart* gambaran umum sistem, *flowchart Text Pre-Processing*, *flowchart Apply* TF-IDF, *flowchart Apply Multinomial Naïve Bayes*, dan *flowchart Testing & Evaluate*.

3.2.1. Gambaran Umum Sistem

Sistem yang dibuat pada penelitian ini merupakan implementasi *Multinomial Naïve Bayes* dan TF-IDF (*Term Frequency - Inverse Document Frequency)* untuk analisis sentimen vaksinasi COVID-19 pada media sosial *twitter*. Penelitian dilakukan dengan mengambil data *tweet* dari *twitter* terlebih dahulu dan kemudian menyimpannya dalam format .csv. Setelah mendapatkan data *tweet*, akan dilakukan *labelling* yang diklasifikasikan menjadi tiga macam, yaitu positif, negatif, atau netral. Label ini jugalah yang nanti akan diprediksi ketika melakukan pengklasifikasian pada *test set*.

Data yang sudah diberikan label akan di-*import* kembali ke dalam bentuk .csv dan dilakukan *text pre-processing*. Tahap ini merupakan tahap yang penting agar data lebih siap dan rapih untuk dilatih. Pada tahap *text pre-processing*, data *tweet* akan dibersihkan, seperti tanda baca, angka, dan mengubah seluruh teks menjadi huruf kecil. Selain itu juga akan ada *tokenization* dimana proses ini digunakan untuk memisahkan sebuah teks menjadi per kata yang disimpan dalam bentuk *array*. Data *tweet* juga akan melalui proses *stopword removal* dan juga *stemming* agar menghilangkan kata-kata yang tidak terlalu memiliki makna penting dan kata-kata yang digunakan dalam teks juga diubah menjadi dalam bentuk kata dasar. Setelah melalui tahap *text pre-processing*, data *tweet* akan dipisahkan antara data untuk *training* dan *testing*. Kedua data *training* dan *testing* akan diterapkan TF-IDF untuk menghitung bobot dari setiap kata dan akan disimpan sebagai model. Model yang berisikan bobot TF-IDF per *term* akan diterapkan untuk pengklasifikasian menggunakan *Multinomial Naïve Bayes*. Kemudian, akan dilakukan uji coba berdasarkan skenario yang dibuat dan mengukur hasil prediksi pada data *testing* menggunakan *confusion matrix* serta terdapat validasi model menggunakan *Stratified K-Fold Cross Validation*.



Gambar 3.1. *Flowchart* Gambaran Umum Sistem

3.2.2. *Crawl Data*

Crawl Data dilakukan dengan menggunakan bantuan *library Twint* dan mencari dengan kata kunci, yaitu ”vaksin”, ”vaksin covid”, dan ”#vaksin” dimulai dari tanggal 13 Januari 2021 hingga 31 Desember 2021.

Crawling menggunakan *library Twint* memiliki batasan terhadap jumlah data *tweet* yang hanya dapat mencapai sekitar 3200 *tweet* paling terakhir. Pada saat *crawling*, perlu dilakukan penggantian tanggal secara per hari. Hal ini dikarenakan ketika *crawling* secara sekaligus dengan meng-*input* tanggal sejak 13 Januari 2021 hingga 31 Desember 2021 tidak dapat dilakukan. Jika meng-*input* langsung, maka hasil yang didapatkan hanyalah *tweet* pada saat tanggal 13 Januari 2021 saja. Oleh karena itu, perlu dilakukan penggantian tanggal secara per hari. Dari hasil *crawling* didapatkan 4418 data *tweet* disimpan dalam format .csv.

3.2.3. *Labelling* Data

Labelling pada data *tweet* sangat dibutuhkan untuk melakukan uji model sehingga mesin dapat mempelajarinya dan mengklasifikasikan label dari data *tweet* yang baru. *Labelling* dapat dilakukan dengan beberapa cara. Dikarenakan ini merupakan *tweet* yang berbahasa Indonesia, *labelling* dapat dilakukan dengan dua cara, yaitu dengan bantuan *library* yang ada pada *python* tetapi perlu menerjemahkan bahasa Indonesia ke bahasa Inggris terlebih dahulu, atau melakukan *manual labelling* dengan cara meminta bantuan dari orang yang kiranya dapat memberikan label secara *manual* terhadap *tweet* yang ada. Pada penelitian ini akan dilakukan *labelling* dengan cara yang kedua, yaitu *manual labelling*.

Labelling dilakukan dengan cara memahami makna dari kalimat yang ada dan kemudian diisikan label sesuai pemahaman yang telah didapat. Pada penelitian ini, label yang diberikan adalah positif, negatif, atau netral. Untuk *labelling* pada data *tweet* akan dilakukan oleh tiga orang yang merupakan kolega penulis, dimana ketiga orang tersebut merupakan mahasiswa dan juga termasuk bagian dari masyarakat umum yang menerima informasi terkait vaksinasi COVID-19. Dari ketiga orang yang memberikan label, akan diambil label manakah yang paling banyak diisi sebagai *final label*-nya. Misalkan pada *tweet* A, orang pertama mengisi 'negatif', orang kedua mengisi 'negatif', dan orang ketiga mengisi 'netral'. Maka *final label*-nya adalah '**negatif**'. Sedangkan jika ketiga orang mengisi label yang berbeda semua, maka *final label* yang akan diambil adalah '**netral**'.

Terdapat petunjuk pengisian pada saat *labelling* yang penulis berikan kepada pengisi. Petunjuk pengisian dapat dilihat pada Tabel 3.1. Indikator untuk menentukan apakah sebuah *tweet* termasuk positif, negatif, atau netral

bereferensikan dari penelitian serupa tetapi penelitian tersebut menganalisis komentar *Instagram* [14].

Tabel 3.1. Tabel Petunjuk Pengisian Label

No.	Petunjuk
1	Baca kalimat pada kolom 'tweet' dan isi apakah kalimat tersebut termasuk label **positif / negatif / netral**.
2	Pastikan pada saat pengisian label, penulisan yang digunakan sama untuk setiap barisnya (tidak ada *typo*, huruf kecil semua).
3	**Positif**: berupa dukungan, saran, bersifat membangun/mendorong, kuota vaksin, dan lain-lain.
4	**Negatif**: tidak percaya, penolakan, fitnah, *hoax*, sindiran, keluhan, tidak menerima vaksin jika belum melihat hasil vaksin dari orang lain.
5	**Netral**: iklan, sapaan, memihak/tidak memihak vaksin, bertanya-tanya terkait vaksin untuk membuat keputusan, cerita diluar vaksin.

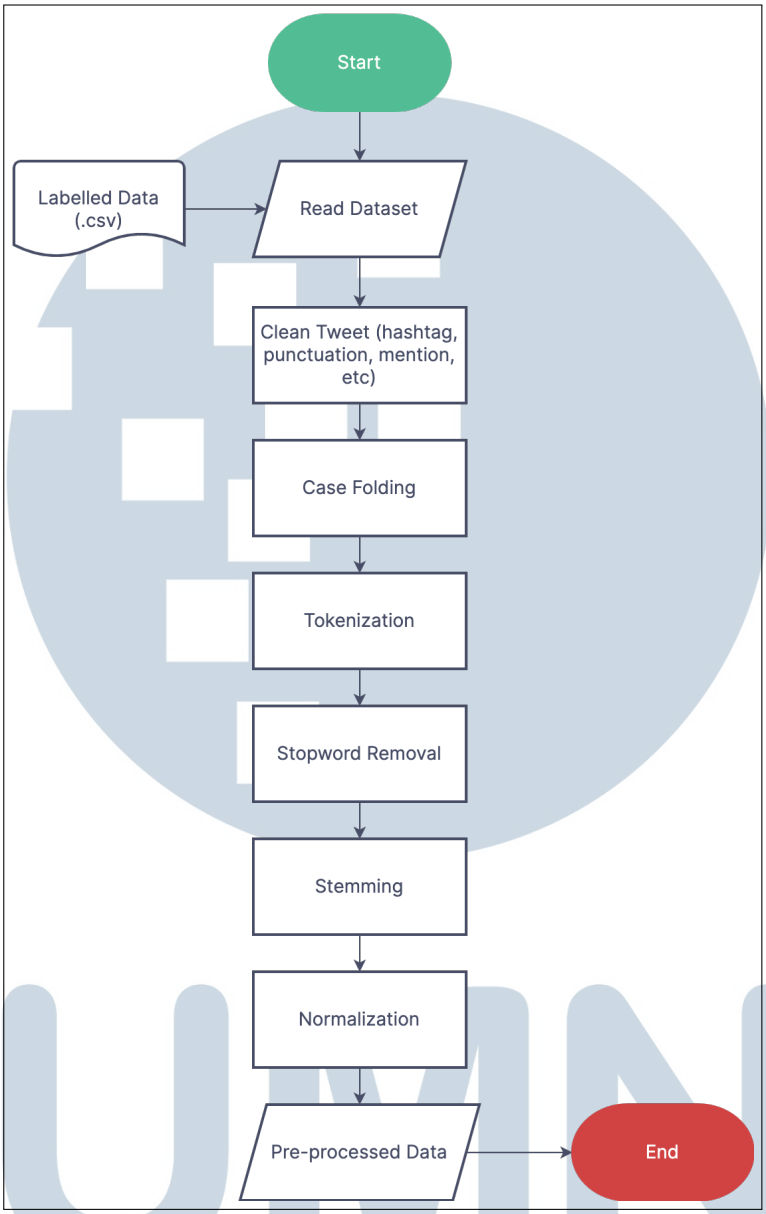
Dataset dari hasil *crawling* yang sebelumnya disimpan dalam bentuk .csv dipindahkan ke media *online* di *google sheets* agar dapat dengan mudah dilakukan *labelling* oleh ketiga kolega penulis. Dari 4418 *tweet* dan hasil *final label* berdasarkan ketiga label yang diisi oleh kolega penulis, didapatkan 1844 *tweet* berlabel positif, 2485 *tweet* berlabel netral, dan 88 *tweet* berlabel negatif. Hasil *labelling* dapat diakses melalui *link* berikut: ***https://docs.google.com/spreadsheets/d/1Z3hMyKl804NE3U6a5n62C_RgXFITQBFmbYL-QuP2DMU/edit?usp=sharing***. Untuk melanjutkan ke tahap selanjutnya, data akan disimpan kembali menjadi dalam bentuk .csv.

3.2.4. *Text Pre-processing*

Gambar 3.2 merupakan alur penerapan *text pre-processing* untuk teks yang berbahasa Indonesia. Data *tweet* yang sudah diberi label akan lanjut ke tahap *text pre-processing*. Pada tahap ini akan dimulai dengan pembacaan data *tweet* terlebih dahulu, kemudian dilakukan pembersihan tanda baca, angka, dan juga menerapkan *case folding* (mengubah seluruh huruf menjadi huruf kecil). Hal ini dilakukan agar tidak terjadinya perbedaan pembacaan kata oleh mesin jika hurufnya tidak disetarakan. Setelah itu, teks pada *tweet* akan diubah menjadi kata-kata yang disimpan dalam

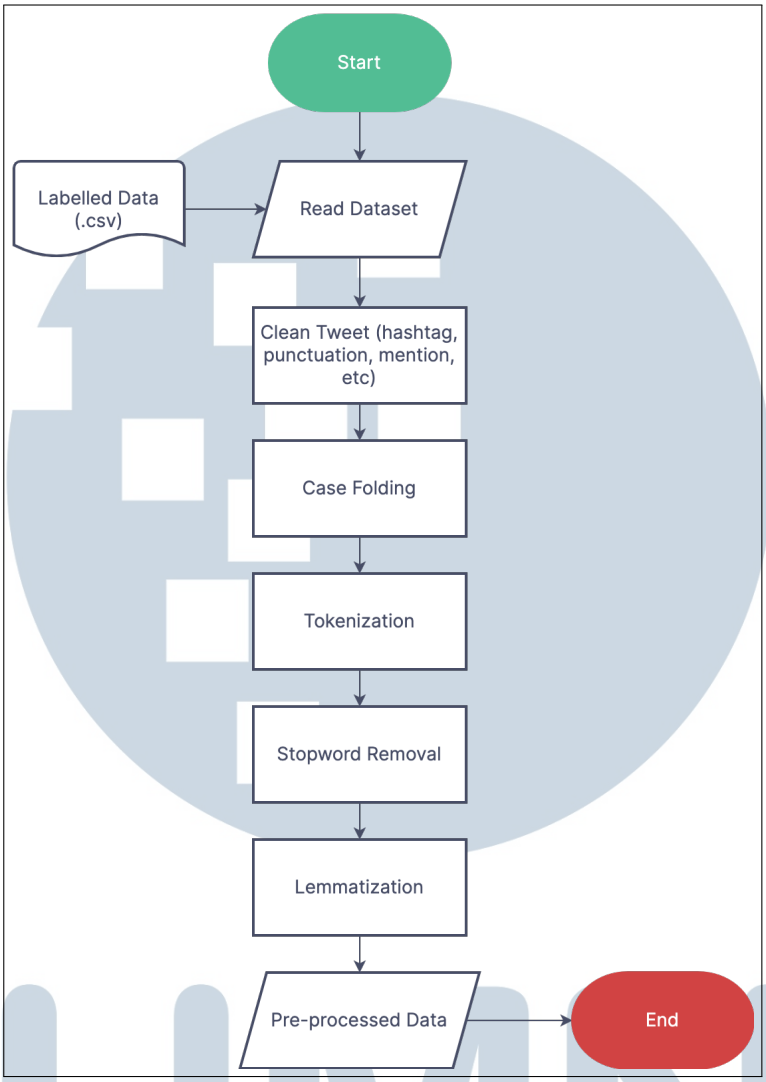
bentuk *array* dengan melakukan *tokenization*.

Selanjutnya adalah *stopword removal*. *Token* yang merupakan hasil dari *tokenization* kemudian akan dibandingan dengan kamus *stopwords* berbahasa Indonesia yang disediakan oleh *library nltk.corpus*. Jika *token* ditemukan pada kamus *stopwords*, maka *token* atau kata tersebut akan dihilangkan. Sedangkan jika tidak ditemukan pada kamus *stopwords*, maka akan dilanjutkan ke proses *normalization*, dimana proses ini adalah mengubah kata-kata yang tidak baku menjadi kata baku atau mengubah kata-kata yang *typo* menjadi kata yang seharusnya. *Normalization* dilakukan secara manual dengan membuat sebuah *dictionary* yang berisikan kata-kata tidak baku dan pasangan kata baku yang seharusnya. Program akan dibuat agar dapat membaca dokumen *dictionary* tersebut dan menggantikan kata-kata yang terdeteksi sebagai kata tidak baku menjadi kata bakunya. Proses selanjutnya adalah *stemming*, dimana *token* hasil normalisasi akan diubah ke dalam bentuk kata dasar. *Stemming* untuk bahasa Indonesia dilakukan dengan memanfaatkan *library* Sastrawi. Kata dasar yang terdapat dalam *Library* Sastrawi sudah dibuat sesuai dengan Kamus Besar Bahasa Indonesia (KBBI). Hasil kata yang telah mencapai tahap *stemming* akan digabungkan kembali menjadi sebuah kalimat utuh kemudian akan disimpan dalam sebagai satu kolom tersendiri sebagai hasil *text pre-processing*.



Gambar 3.2. *Flowchart Text Pre-processing* Bahasa Indonesia

Jika pada gambar sebelumnya, adalah *text pre-processing* untuk teks berbahasa Indonesia, maka selanjutnya pada gambar 3.3 merupakan alur penerapan *text pre-processing* untuk teks berbahasa Inggris.



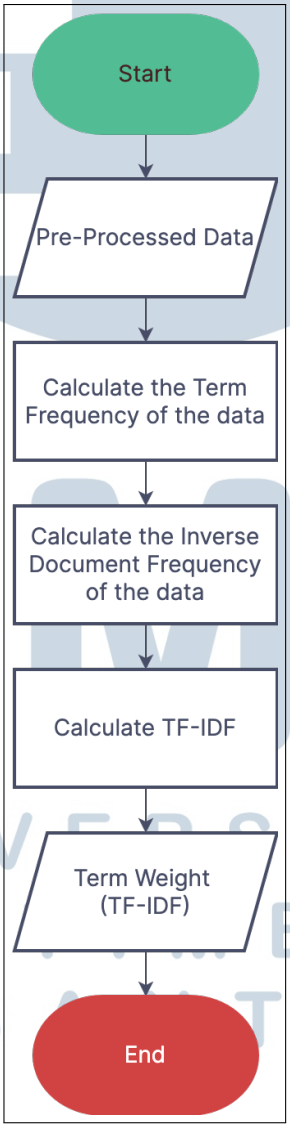
Gambar 3.3. *Flowchart Text Pre-processing* Bahasa Inggris

Text pre-processing untuk data yang berbahasa Inggris dilakukan mengingat adanya beberapa data yang merupakan campuran kalimat antara bahasa Indonesia dan Inggris. Oleh karena itu, data akan diterjemahkan ke bahasa Inggris untuk penyamarataan bahasa. Alur penerapan *text pre-processing* bahasa Inggris ini tidak terlalu berbeda jauh dengan yang bahasa Indonesia. Nantinya hasil dari pre-processing bahasa Indonesia akan digunakan untuk diterjemahkan ke dalam bahasa Inggris yang kemudian akan dilalui kembali proses *cleaning*, *case folding*, *tokenization*, dan *stopword removal*. Setelah melalui keempat tahap tersebut, akan dilanjutkan dengan tahap *lemmatization*. *Lemmatization* ini sebenarnya tidaklah terlalu berbeda jauh dengan *stemming* pada tahap yang ada di *text pre-processing* bahasa In-

donesia. Kedua tahap tersebut sama-sama digunakan untuk mencari kata dasar dari suatu kata yang ada, hanya saja konsep pengubahannya yang berbeda. Jika semua tahap sudah dilalui, maka kata-kata tersebut juga akan digabungkan kembali menjadi kalimat utuh dan disimpan sebagai hasil *text pre-processing* berbahasa Inggris.

3.2.5. *Apply TF-IDF*

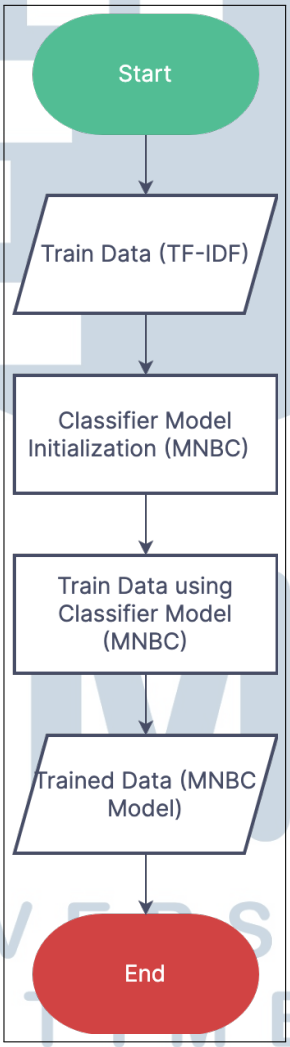
Gambar 3.4 merupakan alur diagram *Apply* TF-IDF. TF-IDF digunakan untuk memberikan bobot pada masing-masing *term* yang dihitung berdasarkan kemunculan *term* pada dokumen dari hasil *text pre-processing*.



Gambar 3.4. *Flowchart Apply* TF-IDF

Pertama-tama, akan dihitung *TF* terlebih dahulu untuk menentukan frekuensi kemunculan *term* menggunakan Persamaan 2.1. Setelah itu, akan dihitung *IDF*-nya menggunakan Persamaan 2.2. Ketika nilai TF dan IDF sudah didapatkan kemudian akan digabung untuk menghasilkan nilai TF-IDF itu sendiri dengan menggunakan Persamaan 2.3. Nilai TF-IDF dari setiap *term* akan disimpan untuk digunakan pada tahap pengklasifikasian.

3.2.6. *Apply Multinomial Naïve Bayes Classifier*



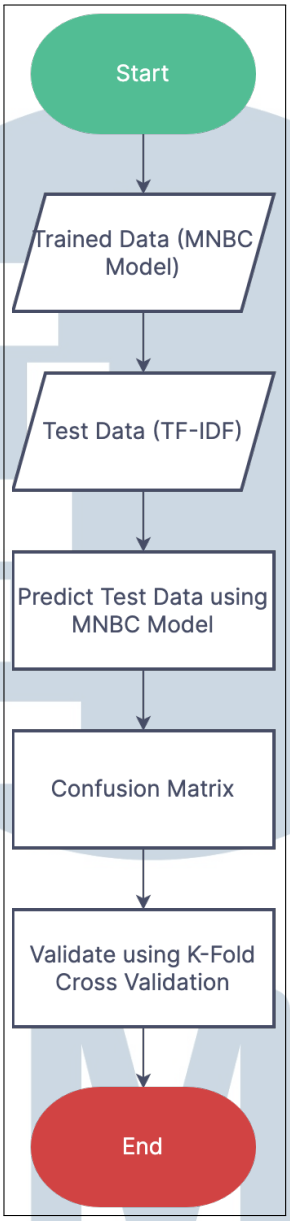
Gambar 3.5. *Flowchart Apply Multinomial Naïve Bayes Classifier*

Tahap selanjutnya adalah melakukan penerapan klasifikasi dengan menggunakan *Multinomial Naïve Bayes Classifier*. Alur penerapannya dapat dilihat pada Gambar 3.5. Dengan menggunakan hasil TF-IDF sebelum-

nya dan data yang telah dipisah antara data *training* dan *testing*, akan dilakukan pelatihan model menggunakan *training data*. Bobot *term* yang didapatkan diterapkan terlebih dahulu pada *training data* dan kemudian dilatih untuk mengklasifikasi label dari *tweet* yang ada. Proses pengklasifikasian data *training* dilakukan dengan menggunakan model klasifikasi dari *MultinomialNB*. Proses ini juga biasanya dikenal sebagai *model fitting*, dimana model *MultinomialNB* akan dilatih mengklasifikasi menggunakan data *training* sehingga bisa digunakan untuk memprediksi data *testing*.

3.2.7. *Testing & Evaluate*

Gambar 3.6 merupakan alur untuk *testing & evaluate*. Setelah melakukan pelatihan model klasifikasi *Multinomial Naïve Bayes* menggunakan data *training*, akan dilanjutkan ke tahap prediksi. Data *testing* akan menuju ke tahap prediksi menggunakan bobot TF-IDF dari data *training* dan model *Multinomial Naïve Bayes* yang sudah dilatih. Kata-kata dari data *testing* yang tidak pernah muncul di data *training*, maka bobot TF-IDF akan dihitung sebagai 0. Data *testing* yang telah diuji dengan berbagai skenario akan dipresentasikan dalam bentuk *confusion matrix* yang dicocokkan dengan label pada data *tweet* aslinya. Hasilnya juga dapat dievaluasi dengan menghitung *accuracy*, *precision*, *recall*, dan juga *f1-score*. Evaluasi akan dilakukan dengan cara membandingkan hasil dari *holdout testing* dengan hasil rata-rata dari validasi menggunakan *Stratified K-Fold Cross Validation*. Cara kerja *holdout testing* sendiri adalah hanya membagi data sekali dengan proporsi yang langusung diinginkan. Sedangkan, cara kerja *Stratified K-Fold Cross Validation* ini adalah dengan membagi *dataset* menjadi dalam sejumlah data *training* sebanyak *k*-1 *fold* dan *testing* untuk satu *fold*-nya secara rata setiap kelas, yang kemudian setelah didapatkan hasilnya akan dihitung rata-rata dari seluruh hasil pengujian sebanyak *k fold* [27].



Gambar 3.6. *Flowchart Testing & Evaluate*