BAB 3
METODOLOGI PENELITIAN

3.1 Metodologi Penelitian

Dalam penelitian yang akan dilakukan, digunakan beberapa metode penelitian sebagai berikut.

1. **Studi Literatur**: Mengumpulkan informasi dari berbagai jurnal penelitian yang berkaitan dengan penelitian yang akan dilakukan.

2. **Pengumpulan Data**: Pada tahap ini, data yang digunakan berasal dari *dataset* Kaggle pada tahun 2023 dengan judul LLM - Detect AI Generated Text Dataset [7]. Melakukan *preprocessing data* untuk memastikan kualitas data, termasuk pembersihan data, dan penanganan data yang memiliki *missing values*.

3. **Perancangan Sistem**: Perancangan sistem dilakukan dengan membuat *flowchart* yang berfungsi sebagai ilustrasi sehingga sistem menjadi terstruktur dan mudah dipahami. Tahap ini akan menggambarkan terkait *preprocessing data* untuk mengubah data ke dalam bentuk yang diinginkan, *feature extraction* untuk mengubah data menjadi format yang dapat digunakan oleh model, *undersampling* untuk menangani ketidakseimbangan data, dan *modelling* yang melibatkan pemilihan serta pelatihan model.

4. **Implementasi Sistem**: Mengimplementasikan model *machine learning* yang telah dipilih menggunakan bahasa pemrograman *Python* dan *library* yang diperlukan. *Dataset* akan dibagi menjadi *train data* dan *test data* untuk melatih dan menguji model. Pada tahap ini, dilakukan penulisan kode untuk *preprocessing data*, pelatihan model, dan evaluasi model.

5. **Pengujian Sistem dan Evaluasi**: Dilakukan untuk menguji dan mengevaluasi model yang sudah dibuat sehingga dapat memberikan informasi sebagai tujuan utama dari penelitian ini. Pengujian sistem menggunakan *dataset* dilakukan dengan menguji data *training* dan *testing*. Selanjutnya, dilakukan pengujian sistem menggunakan teks yang dimasukkan secara manual, yang belum pernah dilihat sebelumnya oleh *model*.

3.2 Teknik Pengumpulan Data

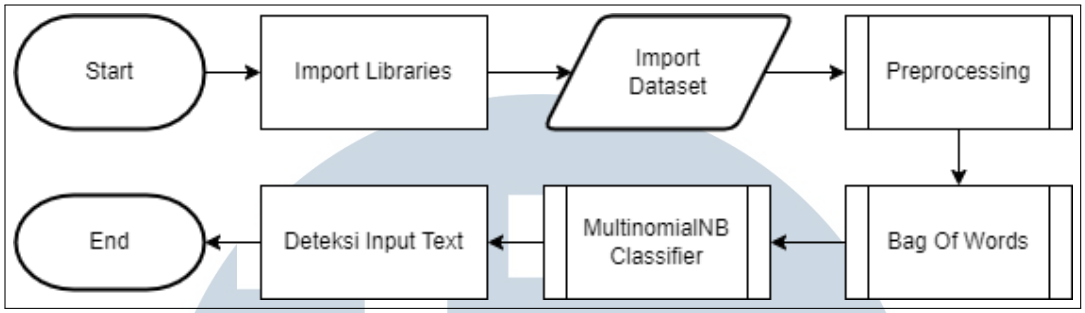
Data yang digunakan pada penelitian ini berasal dari *dataset* Kaggle dengan judul LLM - Detect AI Generated Text Dataset. *Dataset* tersebut terdiri dari 29.145 teks yang dibuat oleh AI dan manusia. Data yang digunakan dalam penelitian ini merupakan data dalam bahasa Inggris karena keterbatasan ketersediaan data dalam bahasa Indonesia. Ketersediaan data dalam bahasa Indonesia terbatas, baik dari segi jumlah maupun variasi, yang membuatnya sulit untuk mendapatkan dataset yang memadai untuk analisis yang komprehensif. Dalam *dataset* ini, tidak dilibatkan seorang pakar dalam pelabelan data, karena *dataset* Kaggle umumnya sudah memiliki label atau metadata yang diperlukan.

3.3 Perancangan Sistem

Perancangan sistem implementasi algoritma *Multinomial Naïve Bayes* untuk deteksi *AI generated text* akan dijabarkan melalui *flowchart* yang terdiri dari *flowchart* utama, *flowchart data preprocessing*, *flowchart module CountVectorizer*, dan *Flowchart Multinomial Naïve Bayes Classifier*.

3.3.1 Flowchart Utama

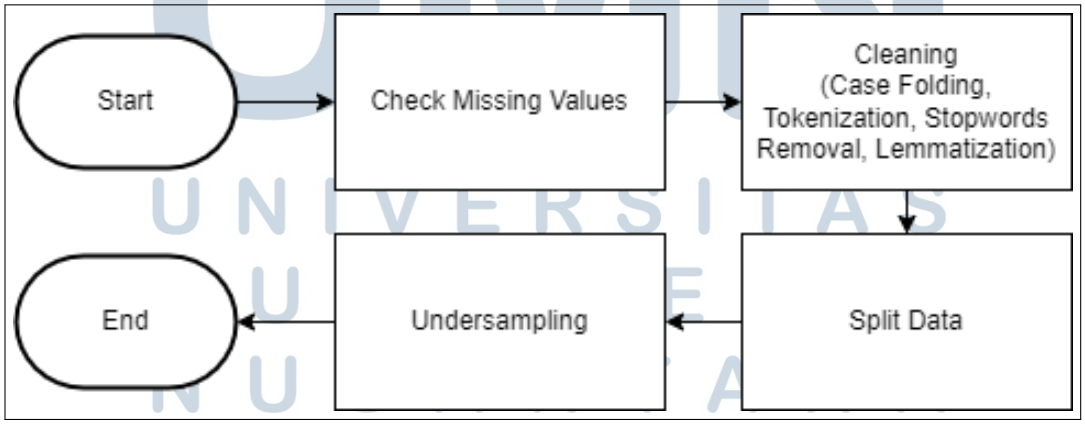
Gambar 3.1 merupakan *flowchart* utama dari proses deteksi *AI generated text*. Proses ini diawali dengan *import library* yang dibutuhkan untuk penelitian, dilanjutkan dengan *import dataset* berbentuk .csv. Setelah melakukan *import dataset* dan *library* yang diperlukan, dilakukan *data preprocessing*. *Data preprocessing* terdiri dari beberapa tahap, yaitu *cleaning*, *case folding*, *tokenization*, *stopwords removal*, *lemmatization*, dan *undersampling*. Tahap selanjutnya adalah menerapkan *feature extraction* BOW, dengan menggunakan *CountVectorizer*, dilanjutkan dengan melakukan klasifikasi menggunakan algoritma *Multinomial Naïve Bayes*. Setelah itu, model akan digunakan untuk melakukan deteksi terhadap *input text*.



Gambar 3.1. *Flowchart* utama

3.3.2 Flowchart Data Preprocessing

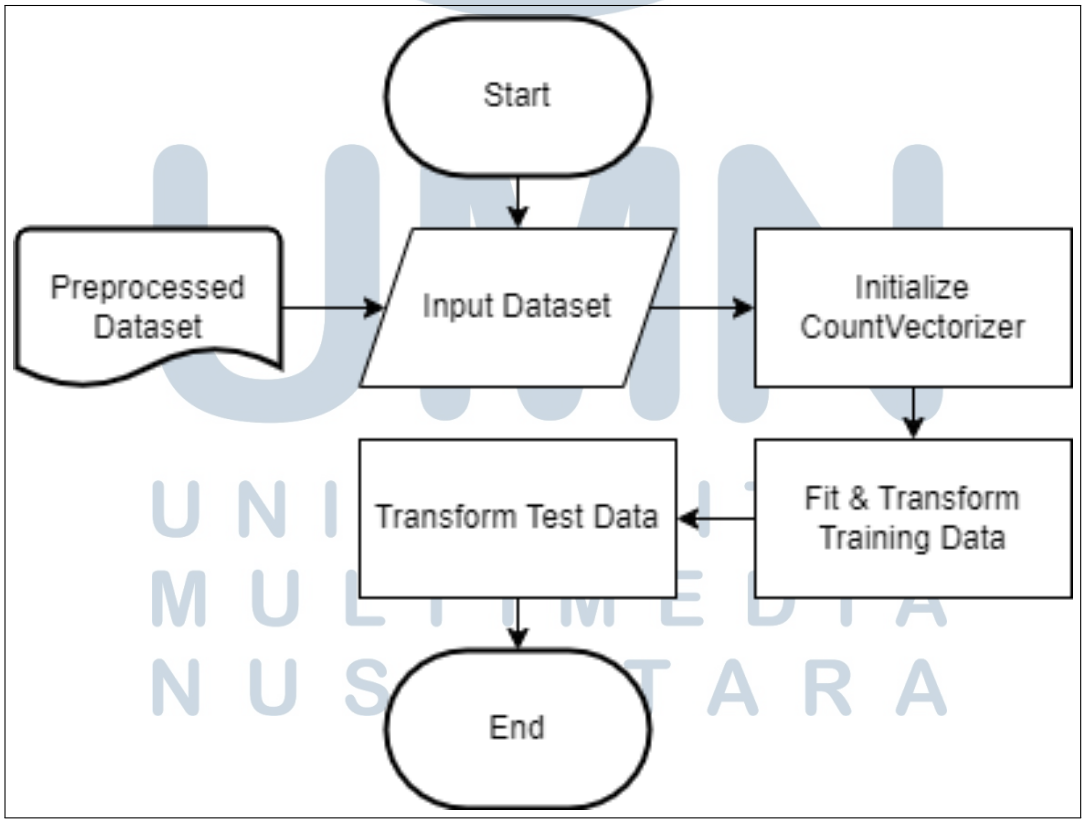
Gambar 3.2 menunjukkan tahapan *data preprocessing*. Pertama, dilakukan validasi *missing values* dengan menggunakan metode *isnull()*. Pada proses *cleaning*, akan diimplementasikan *tokenization*, *case folding*, *stopwords removal*, dan *lemmatization*. Kemudian, dilakukan proses *split data* menggunakan *train_test_split* dari *library sklearn* untuk membagi data menjadi *train data* dan *test data*. Terakhir, diterapkan proses *undersampling* menggunakan *library imblearn.undersampling* untuk mengatasi masalah ketidakseimbangan kelas dalam data, yaitu jumlah sampel dari kelas mayoritas akan dikurangi sehingga menjadi seimbang dengan jumlah sampel dari kelas minoritas, sehingga mencegah model menjadi *bias* terhadap kelas mayoritas.



Gambar 3.2. *Flowchart data preprocessing*

3.3.3 Flowchart Bag of Words

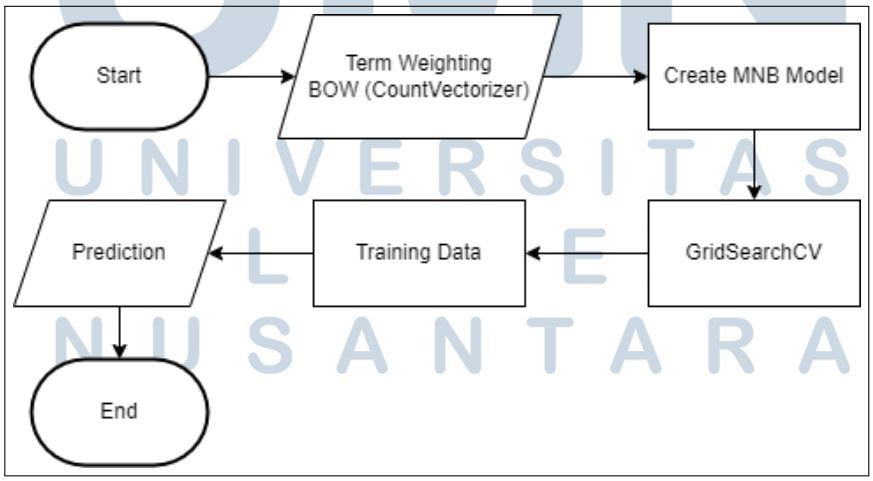
Gambar 3.3 merupakan tahap penerapan *feature extraction* BOW, dengan menggunakan *module CountVectorizer*. Langkah awal dalam proses *feature extraction* BOW menggunakan *CountVectorizer* adalah menginisialisasi objek *CountVectorizer* itu sendiri, dengan melibatkan pembuatan objek baru yaitu *BOW_extractor*. Objek tersebut akan digunakan untuk mengubah teks menjadi representasi numerik yang dapat diproses oleh model. Kemudian, dilakukan proses pelatihan dengan menggunakan *fit* pada objek tersebut, dan memberikan *test data* sebagai argumen. Selama proses ini, *CountVectorizer* akan mempelajari kosakata dari teks yang terdapat dalam *training data* dan membuat matriks *token* hitungan yang merepresentasikan frekuensi kemunculan setiap kata dalam dokumen. Setelah proses pelatihan selesai, dilakukan transformasi *test data* menggunakan kosakata yang telah dipelajari dari *training data* dengan memanggil metode *transform* pada objek dan memberikan *test data* sebagai argumen. Proses ini akan mengubah teks dalam *test data* menjadi matriks *token* hitungan yang sesuai dengan kosakata yang telah dipelajari sebelumnya.



Gambar 3.3. *Flowchart bag of words*

3.3.4 Flowchart Multinomial Naïve Bayes Classifier

Gambar 3.4 merupakan *Flowchart Multinomial Naïve Bayes Classifier*. Dalam tahap ini, data yang telah dipersiapkan dan diubah menjadi vektor *input* melalui proses BOW akan digunakan sebagai *input* untuk melatih model *Multinomial Naïve Bayes Classifier*. Selama proses pelatihan ini, metode GridSearchCV akan digunakan untuk menemukan nilai optimal dari *parameter alpha*, yaitu parameter yang digunakan dalam proses *smoothing* pada model yang bertujuan untuk menangani masalah probabilitas nol dalam perhitungan frekuensi kata yang mungkin tidak muncul pada *train set*. Penggunaan GridSearchCV membantu dalam mencari kombinasi nilai *alpha* yang paling sesuai dengan *training data*, dengan mengujinya menggunakan *cross-validation* yang memungkinkan evaluasi model yang lebih rinci dengan membagi *training data* menjadi beberapa *subset* untuk *train* dan *test* secara bergantian. Kemudian, dilakukan proses pelatihan data menggunakan model yang telah dioptimalkan dengan nilai *alpha* terbaik. Proses ini melibatkan penggunaan *training data* untuk melatih model sehingga model tersebut dapat mengenali pola-pola yang terdapat dalam data. Setelah itu, model digunakan untuk melakukan prediksi pada data dengan mengambil *test data* sebagai *input* dan melakukan klasifikasi pada setiap *test data* untuk memprediksi kelas dari data tersebut. Hasil prediksi dari model akan menjadi *output* dari proses tersebut, yaitu berupa prediksi kelas untuk setiap sampel *test data*. Hasil ini dapat digunakan untuk mengevaluasi performa model.



Gambar 3.4. *Flowchart multinomial naïve bayes classifier*

3.4 Spesifikasi Sistem

Dalam penelitian ini, digunakan *software* dan *hardware* berikut.

1. *Software*:
    - Google Collaboratory
    - Sistem Operasi Windows 11
2. *Hardware*:
    - Intel Core i7-10870H
    - RAM 8GB
    - NVIDIA GeForce GTX 1650