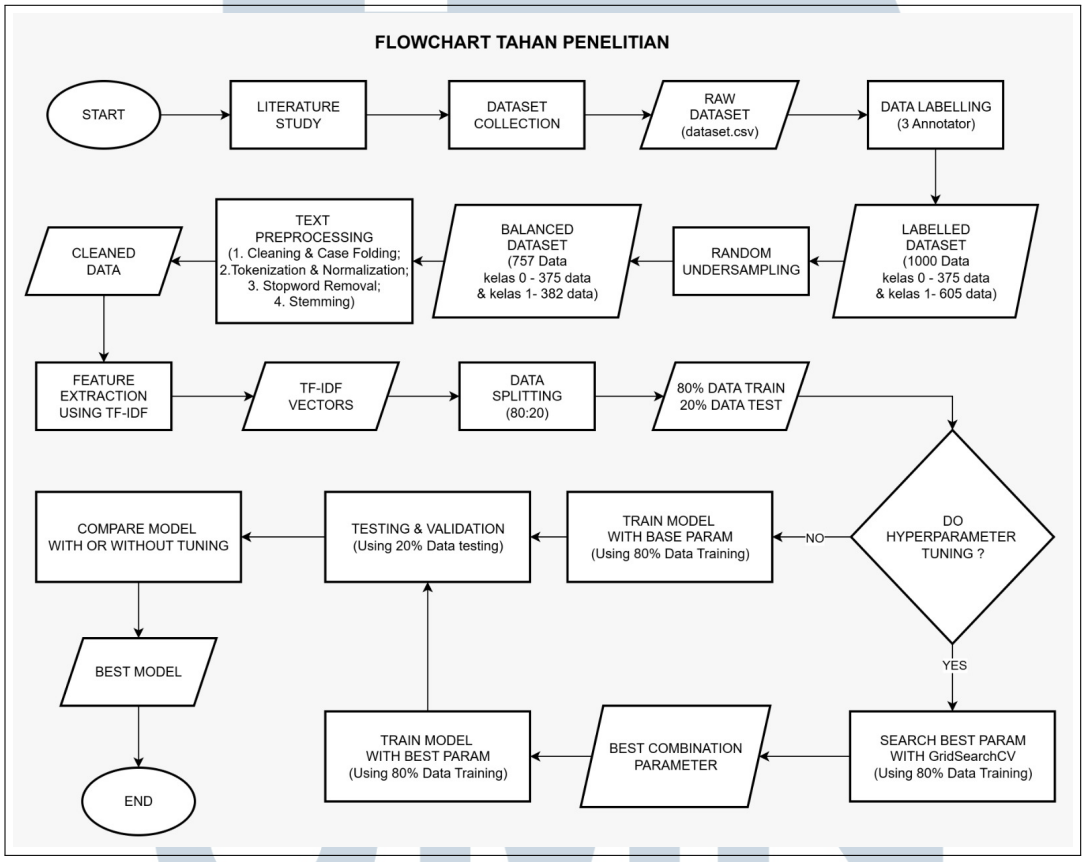
BAB 3
METODOLOGI PENELITIAN

Adapun tahapan penelitian yang dilakukan untuk memahami metode penelitian ini yang dapat dilihat pada diagram flowchart proses metodologi berikut ini:



Gambar 3.1. Diagram tahapan Penelitian

Berdasarkan diagram pada Gambar 3.1, alur penelitian ini dirancang secara sistematis melalui beberapa tahapan utama. Berikut adalah penjelasan rinci untuk setiap tahapan tersebut:

1. Studi Literatur: Tahap awal ini dilakukan untuk mengidentifikasi masalah, merumuskan tujuan penelitian, serta mengkaji teori dan metode yang relevan dari penelitian-penelitian sebelumnya.

2. Pengumpulan dan Pelabelan Data: Proses dimulai dengan pengumpulan data mentah (*RAW DATASET*) melalui teknik *crawling*, menghasilkan sebuah

`dataset.csv`. *Dataset* ini kemudian melalui tahap pelabelan manual oleh tiga anotator. Hasilnya adalah *LABELLED DATASET* yang terdiri dari 1.000 data, dengan distribusi 375 data untuk kelas 0 (*Non Sexual Harassment*) dan 625 data untuk kelas 1 (*Sexual Harassment*).

3. Penyeimbangan Data dengan *Random Undersampling*: Mengingat adanya ketidakseimbangan kelas pada *LABELLED DATASET*, diterapkan teknik *random undersampling* untuk menyeimbangkan distribusi data [22]. Proses ini menghasilkan *BALANCED DATASET* yang terdiri dari total 757 data, dengan rincian 375 data untuk kelas 0 dan 382 data untuk kelas 1.

4. *Text Preprocessing*: *Dataset* yang telah seimbang kemudian melewati serangkaian tahap *text preprocessing* untuk pembersihan dan standardisasi. Tahapan ini meliputi: (1) *Cleaning & Case Folding*, (2) *Tokenization & Normalization*, (3) *Stopword Removal*, dan (4) *Stemming*. Output dari tahap ini adalah *CLEANED DATA*.

5. *Feature Extraction*: *CLEANED DATA* yang sudah bersih kemudian diubah menjadi representasi vektor numerik. Proses *feature extraction* ini menggunakan metode *Term Frequency-Inverse Document Frequency* (TF-IDF), yang menghasilkan *TF-IDF VECTORS* [29].

6. *Data Splitting*: *TF-IDF VECTORS* kemudian dibagi menjadi data latih dan data uji dengan rasio 80:20. Alokasi ini menghasilkan *80% DATA TRAIN* untuk melatih model dan *20% DATA TEST* untuk menguji model [14] [17] [41].

7. Pemodelan dan Eksperimen Komparatif: Pada tahap ini, alur dipecah menjadi dua skenario berdasarkan keputusan ”DO HYPERPARAMETER TUNING?”.

    - Jalur ”NO” (Model *Baseline*): Model *Random Forest* dilatih menggunakan parameter dasar (*base param*) pada *80% DATA TRAIN*.
    - Jalur ”YES” (Model *Tuning*): Pertama, dilakukan pencarian parameter terbaik menggunakan *GridSearchCV* pada *80% DATA TRAIN*. Proses ini menghasilkan *BEST COMBINATION PARAMETER*, yang kemudian digunakan untuk melatih model *Random Forest* kedua.

8. Pengujian dan Validasi: Kedua model yang telah dilatih (model *baseline* dan model hasil *tuning*) kemudian diuji dan divalidasi performanya secara terpisah menggunakan *20% DATA TEST* yang sama.

9. Perbandingan Model dan Kesimpulan: Tahap terakhir adalah membandingkan kinerja dari kedua model (*COMPARE MODEL WITH OR WITHOUT TUNING*). Model dengan performa tertinggi akan ditetapkan sebagai *BEST MODEL*, yang menjadi hasil akhir dari penelitian sebelum proses diakhiri (*END*).

Tahapan-tahapan tersebut dirancang secara sistematis agar model yang dihasilkan mampu mendeteksi komentar pelecehan seksual verbal dengan tingkat akurasi yang optimal. Proses evaluasi pada tahap akhir menjadi tolak ukur utama dalam menilai performa model sebelum digunakan secara lebih luas.

3.1 Spesifikasi sistem yang Digunakan

Penelitian ini dilakukan menggunakan perangkat dengan spesifikasi perangkat keras dan perangkat lunak tertentu untuk mendukung proses pengolahan data dan pembangunan model deteksi komentar pelecehan seksual verbal. Bahasa pemrograman yang digunakan adalah Python dengan lingkungan pengembangan berbasis Jupyter Notebook, yang mendukung eksplorasi data secara interaktif serta memudahkan implementasi berbagai pustaka *machine learning*. Proses *data crawling* dilakukan menggunakan alat bernama *tweet harvest* yang dijalankan di lingkungan Google Colab dengan dukungan `Node.js` untuk menjalankan skrip *scraping* secara otomatis.

Adapun pustaka (*libraries*) utama yang digunakan meliputi pustaka untuk *data preprocessing*, *feature extraction*, pembangunan model, serta evaluasi model. Spesifikasi perangkat keras seperti RAM 8GB dan prosesor Intel i5 dipilih untuk memastikan model *Random Forest* yang dibangun, terutama dengan proses *ensemble learning* yang melibatkan banyak pohon keputusan, dapat berjalan secara optimal. Selain itu, penggunaan Google Colab mendukung proses *scraping* yang membutuhkan akses jaringan serta menyediakan lingkungan komputasi yang telah terkonfigurasi dengan baik.

Dengan spesifikasi sistem ini, seluruh tahapan penelitian mulai dari pengumpulan data hingga evaluasi model dapat berjalan secara efisien, mendukung proses pembangunan model *Random Forest* yang menjadi fokus utama penelitian.

Rincian spesifikasi sistem yang digunakan dalam penelitian ini disajikan pada Tabel 3.1.

Tabel 3.1. Spesifikasi Sistem yang Digunakan

Komponen	Spesifikasi
Processor	Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz, 2496 MHz, 4 Core(s), 8 Logical Processor(s)
RAM	8.00 GB
Operating System	Microsoft Windows 11 Home Single Language
Programming Language	Python
IDE	Jupyter Notebook
Environment	Google Colab
Main Libraries	`pandas`, `NumPy`, `scikit-learn`, `matplotlib`, `seaborn`, `NLTK`, `Sastrawi`, `TfidfVectorizer`
Data Crawling Tools	`Tweet-Harvest`, `Node.js`

3.2 Studi Literatur

Studi literatur dilakukan sebagai langkah awal dalam penelitian ini untuk memahami konteks permasalahan serta merumuskan latar belakang, rumusan masalah, dan tujuan penelitian. Proses ini dimulai dengan menelusuri penelitian yang membahas peran media sosial dalam kehidupan remaja, termasuk dampak positif dan negatif yang ditimbulkan. Dari hasil penelusuran, ditemukan bahwa media sosial memiliki peran signifikan dalam membentuk interaksi sosial remaja, namun di sisi lain juga membawa risiko, seperti meningkatnya kasus kekerasan berbasis gender secara daring (KBGO) [6].

Tahap selanjutnya adalah menelaah literatur yang secara khusus membahas dampak negatif media sosial terhadap remaja, terutama yang berkaitan dengan aspek gender. Penelitian-penelitian tersebut mengungkap bahwa kelompok remaja perempuan lebih rentan menjadi korban pelecehan seksual di platform media sosial [6]. Untuk mendukung temuan ini, dikumpulkan pula data mengenai kasus kekerasan berbasis gender daring dari berbagai sumber yang relevan. Data tersebut digunakan untuk memperkuat latar belakang penelitian dan menyoroti urgensi perlunya pengembangan model deteksi guna meminimalisasi dampak psikologis pada korban.

Selain memahami dampak sosial, dilakukan pula penelusuran terhadap

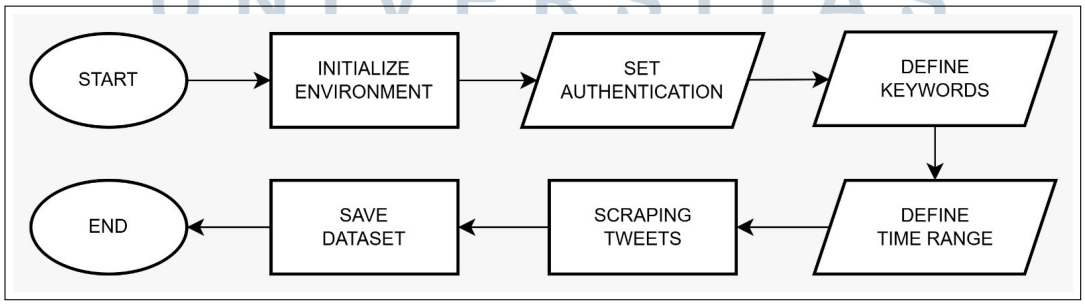
metode deteksi komentar negatif berbasis gender yang telah dikembangkan dalam penelitian sebelumnya. Berbagai penelitian telah mengembangkan model deteksi menggunakan beragam algoritma *machine learning*, seperti *Support Vector Machine* (SVM) dan *Naïve Bayes* [9]. Namun, hasil penelitian tersebut membuka peluang untuk mengeksplorasi cakupan deteksi yang lebih spesifik yaitu komentar pelecehan seksual.

Berdasarkan kajian lebih mendalam, perhatian diarahkan pada algoritma *Random Forest* karena beberapa penelitian terdahulu menunjukkan performa yang baik dalam mendeteksi komentar negatif berbasis lain, seperti ujaran kebencian dan *cyberbullying* [14]. *Random Forest* menawarkan keunggulan dalam mengurangi risiko *overfitting* dengan memanfaatkan konsep *ensemble learning* melalui penggabungan beberapa pohon keputusan untuk menghasilkan prediksi yang lebih stabil dan akurat [15].

Dengan demikian, studi literatur ini memberikan landasan yang kuat bagi pengembangan model deteksi komentar pelecehan seksual verbal di media sosial X menggunakan algoritma *Random Forest*. Pengetahuan yang diperoleh dari studi ini tidak hanya membantu dalam merumuskan permasalahan dan tujuan penelitian, tetapi juga menjadi pijakan utama dalam merancang metode yang diterapkan pada penelitian ini.

3.3 Data Collection

Pada penelitian ini, data dikumpulkan dari media sosial X (sebelumnya dikenal sebagai Twitter) menggunakan teknik *web scraping* untuk memperoleh komentar yang mengandung pelecehan seksual verbal berbahasa Indonesia. Proses pengumpulan data dilakukan dalam beberapa tahap, dimulai dari penyiapan lingkungan kerja hingga penyimpanan *dataset*, yang secara rinci dapat dilihat pada flowchart berikut ini.



Gambar 3.2. Flowchart Data Collection

Sumber data berasal dari media sosial X dengan rentang waktu pengambilan mulai dari 1 September 2023 hingga 1 Januari 2024. Rentang waktu ini dipilih merujuk pada temuan Tim Peneliti Monash University yang mengidentifikasi 3.528 unggahan dan komentar bermuatan seksual atau vulgar pada periode tersebut [7].

Untuk memastikan data yang dikumpulkan relevan dengan konteks pelecehan seksual, digunakan kata kunci yang dipilih berdasarkan penelitian terdahulu yang telah mengidentifikasi istilah-istilah yang sering muncul dalam kasus pelecehan seksual daring. Adapun kata kunci yang digunakan meliputi "jalang" [18]; "lonte","leceh","tobrut","pelecehan", dan "pelecehan seksual" [19]; "pelacur" dan "digilir" [20];"perkosa" dan "jablay" [21]; "janda" [47]. Pemilihan kata kunci ini bertujuan untuk memastikan data yang diambil benar-benar mencerminkan komentar berunsur pelecehan seksual verbal di media sosial X.

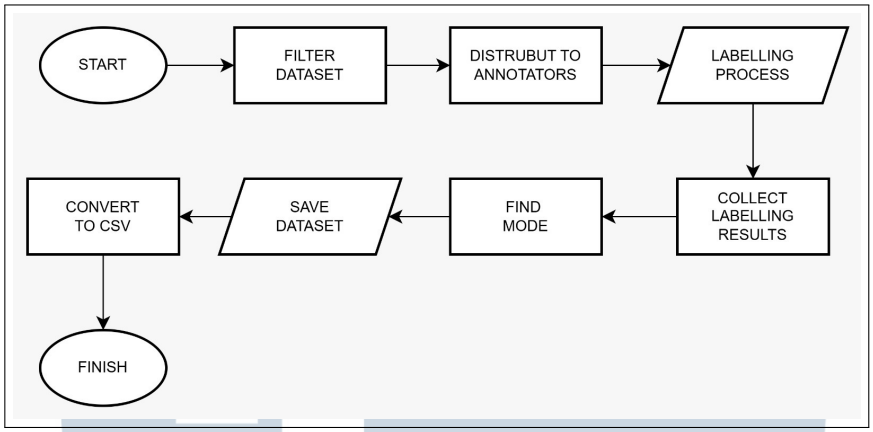
Proses *crawling* dilakukan menggunakan alat bernama *tweet-harvest* yang dijalankan di lingkungan Google Colab dengan bahasa pemrograman Python. Sebelum melakukan *scraping*, dilakukan proses autentikasi menggunakan token otorisasi Twitter agar dapat mengakses API yang digunakan untuk mengambil data. Setelah autentikasi berhasil, proses *crawling* dimulai dengan memasukkan kata kunci yang telah ditentukan untuk menyaring komentar yang mengandung unsur pelecehan seksual [25].

Setelah data berhasil dikumpulkan, hasil *scraping* disimpan dalam format CSV untuk memudahkan proses analisis lebih lanjut. Jumlah total data mentah yang berhasil dikumpulkan dari proses ini adalah sebanyak 1.000 komentar. *Dataset* mentah inilah yang kemudian menjadi input utama untuk tahap selanjutnya, yaitu proses pelabelan data.

3.4 Data Labelling

Setelah proses pengumpulan data selesai, tahap selanjutnya adalah memberikan label pada setiap komentar untuk menentukan apakah komentar tersebut mengandung pelecehan seksual atau tidak. Proses pelabelan dilakukan secara manual dengan melibatkan tiga mahasiswa yang memiliki kualifikasi minimal nilai A- dalam mata kuliah Bahasa Indonesia [48]. Kriteria ini dipilih untuk memastikan bahwa *annotator* memiliki kemampuan yang memadai dalam memahami konteks bahasa dari komentar yang dianalisis. Selain itu, keterlibatan tiga *annotator* bertujuan untuk meminimalkan bias subjektif dan meningkatkan objektivitas melalui metode *voting* [48]. Tahapan pelabelan ini dilakukan secara

sistematis sesuai dengan alur yang digambarkan pada flowchart berikut:



Gambar 3.3. Flowchart Data Labelling

Berdasarkan flowchart di atas, berikut adalah penjelasan rinci untuk setiap tahapan pelabelan data yang dilakukan:

1. Filter Dataset: Proses diawali dengan memfilter *dataset* mentah hasil *scraping* untuk hanya mengambil kolom yang berisi teks komentar yang akan dianalisis.

2. Distribusi dan Proses Pelabelan: *Dataset* yang sudah difilter kemudian didistribusikan kepada tiga *annotator* melalui Google Sheets untuk dilabeli secara independen. Untuk menjaga konsistensi, para *annotator* mengikuti pedoman pelabelan yang telah ditentukan:

    - Label *1 (Sexual Harassment)* diberikan jika sebuah komentar secara eksplisit mengandung satu atau lebih kata kunci yang telah ditetapkan (contoh: “jalang”, “lonte”, “perkosa”, dll.) [18][19][20][21][47].
    - Label *0 (Non Sexual Harassment)* diberikan untuk komentar yang tidak mengandung kata kunci tersebut atau konteksnya tidak mengarah pada pelecehan seksual verbal akan diberi label 0 (*Non Sexual Harassment*).

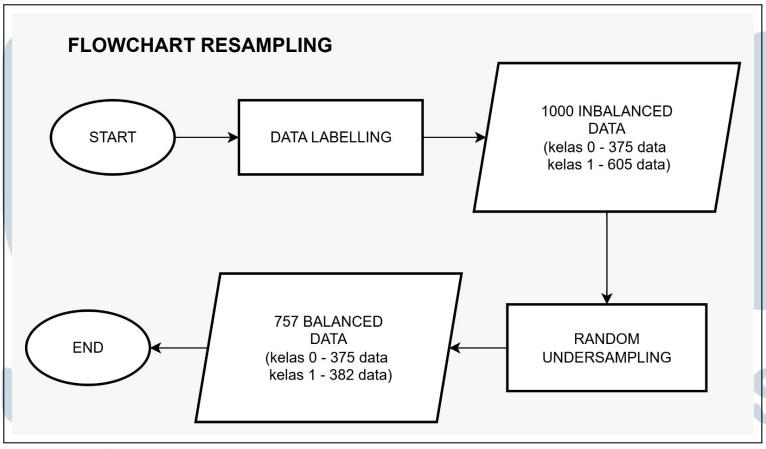
3. Agregasi Hasil dengan Modus: Setelah proses pelabelan selesai, hasil dari ketiga *annotator* dikumpulkan (*Collect Labelling Results*). Untuk menentukan label akhir dari setiap komentar, digunakan metode modus (*Find Mode*), yaitu memilih label yang paling sering muncul di antara ketiga *annotator*. Metode ini dipilih karena secara efektif merepresentasikan suara mayoritas dan meningkatkan objektivitas pelabelan [48].

4. Penyimpanan dan Konversi Dataset: *Dataset* yang telah memiliki label akhir kemudian disimpan (*Save Dataset*) dalam format `.xlsx` untuk validasi dan pemeriksaan ulang. Setelah divalidasi, file tersebut dikonversi ke format `.csv` (*Convert to CSV*) agar siap digunakan sebagai *dataset* final untuk tahap *text preprocessing*.

Dengan tahapan ini, dihasilkan *dataset* yang sudah dilabeli secara akurat dan objektif sesuai dengan ruang lingkup penelitian.

3.5 Random Undersampling

Setelah melalui tahap pelabelan, *dataset* yang dihasilkan menunjukkan adanya masalah kelas tidak seimbang (*imbalanced class*), di mana jumlah data untuk setiap kelas tidak terdistribusi secara merata. Masalah seperti ini umum terjadi dan dapat menyebabkan model klasifikasi menjadi bias atau cenderung berpihak pada kelas mayoritas, sehingga menurunkan performa prediksinya [22]. Untuk mengatasi tantangan ini, diterapkan teknik *resampling*, khususnya *random undersampling*, yang terbukti efektif dalam meningkatkan akurasi dibandingkan metode lain seperti *oversampling* [22] [27]. Alur kerja dari proses ini dapat dilihat pada Gambar 3.4.



Gambar 3.4. Flowchart Random Undersampling

Berdasarkan flowchart di atas, berikut adalah penjelasan rinci untuk setiap tahapan *random undersampling* yang dilakukan:

1. Input Dataset Tidak Seimbang: Proses dimulai dengan *dataset* hasil pelabelan yang terdiri dari total 1.000 data. Dataset ini memiliki distribusi kelas

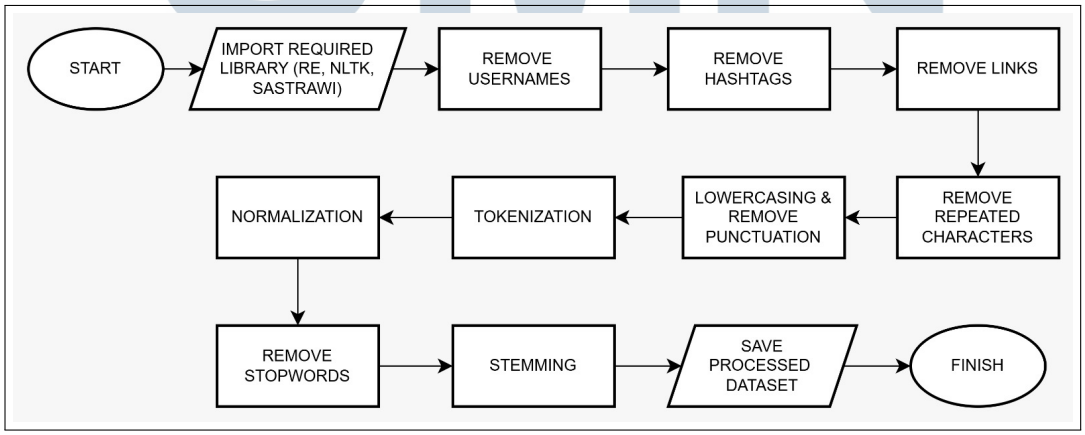
yang tidak seimbang, dengan rincian 375 data untuk kelas 0 (*Non Sexual Harassment*) yang merupakan kelas minoritas, dan 605 data untuk kelas 1 (*Sexual Harassment*) sebagai kelas mayoritas.

2. Penerapan *Random Undersampling*: Teknik *random undersampling* kemudian diterapkan secara spesifik pada kelas mayoritas (kelas 1). Proses ini tidak bertujuan untuk menciptakan rasio 1:1 yang kaku, melainkan untuk mengurangi tingkat ketimpangan secara proporsional [22]. Pengurangan sampel dilakukan secara acak, namun tetap dengan pertimbangan untuk mempertahankan data-data yang paling relevan dan representatif dari kelas mayoritas, sehingga informasi yang krusial tidak ikut terbuang [22][27]. Pendekatan ini memastikan *dataset* menjadi lebih seimbang tanpa mengorbankan kualitas data secara signifikan [22].

3. Output Dataset Seimbang: Setelah proses *undersampling*, dihasilkan sebuah *dataset* baru yang lebih seimbang. *Dataset* akhir ini terdiri dari total 757 data, dengan komposisi 375 data untuk kelas 0 (*Non Sexual Harassment*) dan 382 data untuk kelas 1 (*Sexual Harassment*. *Dataset* inilah yang akan digunakan untuk tahap *preprocessing* dan pemodelan selanjutnya.

3.6 Text Preprocessing

Setelah pelabelan, data melalui tahap *text preprocessing* untuk dibersihkan sebelum klasifikasi. Proses ini menggunakan pustaka `re` untuk pencocokan pola dengan ekspresi reguler, `NLTK` untuk tokenisasi dan penghapusan *stopwords*, serta `Sastrawi` untuk *stemming*. Alur proses ditampilkan pada flowchart berikut:



Gambar 3.5. Flowchart Text Preprocessing

Berdasarkan flowchart di atas, berikut adalah penjelasan rinci untuk setiap tahapan *text preprocessing* yang dilakukan:

1. Pembersihan Awal (*Initial Cleaning*): Tahap ini fokus pada penghapusan elemen-elemen non-tekstual yang tidak relevan. Proses ini mencakup:

   - *Remove Usernames*: Menghapus *mention* atau nama pengguna yang diawali dengan simbol .
   - *Remove Hashtags*: Menghapus tanda pagar (#) beserta kata yang mengikutinya.
   - *Remove Links*: Menghapus tautan atau URL dari dalam teks komentar.

   Ketiga proses ini bertujuan untuk membersihkan teks dari identitas pengguna dan elemen eksternal yang tidak berkontribusi pada analisis makna komentar [9].

2. Penghapusan Karakter Berulang (*Remove Repeated Characters*): Proses ini bertujuan untuk menormalisasi kata-kata yang mengandung pengulangan huruf secara berlebihan (contoh: ”kasarrrr” menjadi ”kasar”). Langkah ini penting untuk menjaga konsistensi kosakata yang akan dianalisis oleh model [14].

3. *Lowercasing* & *Remove Punctuation*: Seluruh teks diubah menjadi format huruf kecil (*lowercase*) untuk menyeragamkan kata. Selain itu, semua angka dan tanda baca juga dihapus karena tidak memiliki relevansi dalam konteks deteksi pelecehan verbal [9].

4. Tokenisasi (*Tokenization*): Setelah teks bersih, dilakukan proses tokenisasi untuk memecah kalimat menjadi unit-unit kata individual yang disebut ”token”. Tahap ini merupakan langkah fundamental untuk memungkinkan analisis pada level kata [16].

5. Normalisasi Kata (*Normalization*): Tahap ini berfungsi untuk mengubah kata-kata tidak baku atau singkatan (*slang*) ke dalam bentuk bakunya (contoh: ”yg” menjadi ”yang”). Proses ini dilakukan menggunakan kamus normalisasi yang diadaptasi dari beberapa penelitian terdahulu telah teruji efektif untuk digunakan dalam proses klasifikasi [14][49][50].

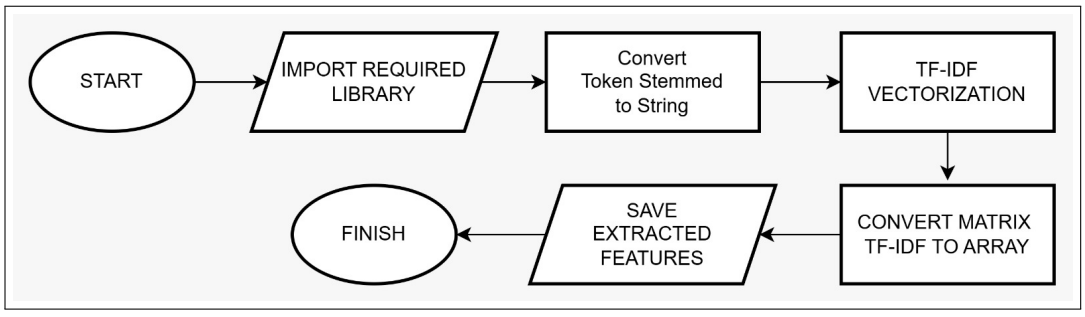
6. Penghapusan *Stopwords* (*Remove Stopwords*): Kata-kata umum yang tidak memiliki makna signifikan dalam analisis (seperti ”dan”, ”di”, ”yang”) dihapus dari setiap komentar. Daftar *stopwords* yang digunakan merupakan gabungan dari pustaka `NLTK` dan tambahan kata custom yang disesuaikan dengan konteks penelitian [14][16].

7. *Stemming*: Tahap terakhir adalah mengubah setiap kata ke dalam bentuk kata dasarnya dengan menghilangkan semua imbuhan. Proses ini menggunakan pustaka `Sastrawi` yang dirancang khusus untuk Bahasa Indonesia, sehingga kata-kata dengan variasi imbuhan dapat dikenali sebagai satu entitas yang sama oleh model [14].

Setelah seluruh tahapan *preprocessing* selesai, *dataset* yang telah bersih dan terstruktur disimpan dalam format CSV untuk digunakan pada tahap *feature extraction*.

3.7 Feature Extraction

Tahap *feature extraction* dalam penelitian ini menggunakan *Term Frequency-Inverse Document Frequency* (TF-IDF) untuk mengubah teks hasil *preprocessing* menjadi representasi numerik. Metode ini dipilih karena pola bahasa dalam komentar pelecehan seksual cenderung bervariasi dan kontekstual [12]. Dengan pendekatan TF-IDF, kata-kata yang secara spesifik menunjukkan indikasi pelecehan seksual dapat memiliki bobot lebih tinggi, sehingga membantu model membedakan komentar ofensif dari yang netral. Berikut disajikan diagram alir (*flowchart*) proses *feature extraction* yang menggambarkan bagaimana teks yang sudah diproses diubah menjadi fitur numerik.



Gambar 3.6. Flowchart Feature Extraction

Berdasarkan flowchart di atas, berikut adalah penjelasan rinci untuk setiap tahapan *feature extraction* yang dilakukan:

1. Impor Pustaka (*Import Required Library*): Langkah pertama adalah mengimpor semua pustaka Python yang diperlukan untuk proses ini, terutama *library* `scikit-learn` yang menyediakan fungsionalitas untuk vektorisasi TF-IDF [28].

2. Konversi Token ke String (*Convert Token Stemmed to String*): Data hasil akhir dari tahap *preprocessing* yang berbentuk daftar token (kata-kata terpisah) digabungkan kembali menjadi satu kalimat utuh (format *string*). Langkah ini krusial karena model `TfidfVectorizer` dari `scikit-learn` memerlukan input berupa teks dalam format *string*, bukan daftar token [28][31].

3. Vektorisasi TF-IDF (*TF-IDF Vectorization*): Pada tahap inti ini, setiap komentar dalam format *string* dikonversi menjadi representasi vektor numerik [28]. `TfidfVectorizer` akan menghitung bobot TF-IDF untuk setiap kata unik di seluruh korpus, yang merepresentasikan tingkat kepentingan kata tersebut dalam sebuah komentar [29].

4. Konversi Matriks ke Array (*Convert Matrix TF-IDF to Array*): Hasil dari vektorisasi TF-IDF adalah sebuah matriks renggang (*sparse matrix*). Untuk memudahkan proses komputasi pada algoritma *machine learning*, matriks ini kemudian diubah menjadi format *array* yang lebih padat (*dense array*) [28][31].

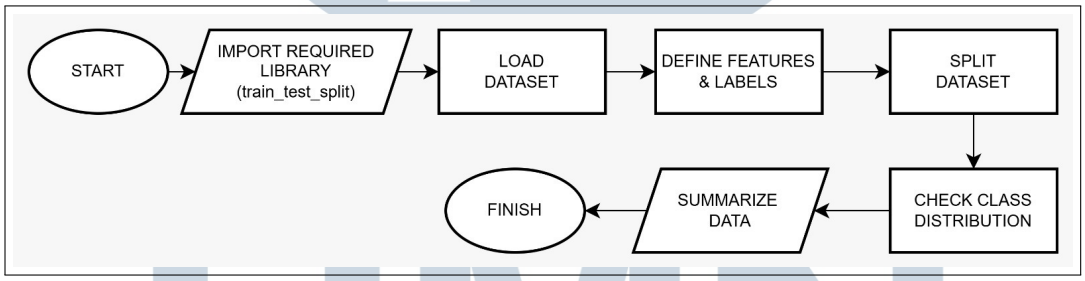
5. Simpan Fitur Hasil Ekstraksi (*Save Extracted Features*): Vektor numerik yang merupakan representasi akhir dari setiap komentar disimpan sebagai *dataset* fitur. *Dataset* inilah yang akan digunakan sebagai input pada tahap pemodelan dan klasifikasi selanjutnya.

Meskipun proses penggunaan TF-IDF menghasilkan dimensi fitur yang besar akibat tidak adanya pembatasan `max_features` guna memastikan seluruh kata unik dipertahankan agar model dapat menangkap pola secara maksimal, algoritma *Random Forest* tetap mampu mengatasi tingginya jumlah fitur dengan mekanisme pemilihan fitur secara acak pada setiap pohon keputusan. Dengan demikian, model tidak perlu menggunakan seluruh fitur secara bersamaan dalam setiap *Decision Tree* yang dibangun, sehingga risiko *overfitting* dapat diminimalkan [15].

Jika proses *feature extraction* ini tidak dilakukan, model akan mengalami kesulitan dalam mengenali pola linguistik dalam komentar pelecehan seksual. Algoritma *machine learning* tidak akan dapat membedakan kata-kata dengan makna pelecehan dan kata-kata umum dalam bahasa Indonesia tanpa representasi numerik yang tepat. Hal ini dapat menyebabkan model gagal mendeteksi komentar yang mengandung pelecehan secara akurat, sehingga performa klasifikasi menjadi tidak optimal [9][24].

3.8 Data Splitting

Setelah proses *feature extraction* selesai, langkah selanjutnya adalah membagi *dataset* menjadi dua bagian terpisah, yaitu data latih (*training data*) dan data uji (*testing data*) [32]. Proses ini bertujuan agar model yang dibangun dapat dilatih secara efektif menggunakan mayoritas data, kemudian kinerjanya dievaluasi secara objektif pada data yang belum pernah dilihat sebelumnya. Tahapan dari proses *data splitting* ini diilustrasikan pada gambar berikut:



Gambar 3.7. Flowchart Data Splitting

Berdasarkan flowchart di atas, berikut adalah penjelasan rinci untuk setiap tahapan *data splitting* yang dilakukan:

1. Impor Pustaka (*Import Required Library*): Tahap pertama adalah mengimpor fungsi `train_test_split` dari pustaka `sklearn.model_selection`, yang merupakan alat utama untuk proses pemisahan data.
2. Memuat Dataset (*Load Dataset*): Selanjutnya, *dataset* yang telah melalui tahap *feature extraction* dan sudah berbentuk vektor numerik dimuat ke dalam lingkungan kerja untuk diproses.
3. Mendefinisikan Fitur dan Label (*Define Features & Labels*): Pada tahap ini, *dataset* dipisahkan menjadi dua komponen utama: variabel independen (X)

yang berisi kumpulan vektor TF-IDF, dan variabel dependen (y) yang berisi label kelas (0 untuk *Non Sexual Harassment* dan 1 untuk *Sexual Harassment*).

4. Pemisahan Dataset (*Split Dataset*): Data kemudian dipisah menjadi 80% data latih dan 20% data uji. Pemilihan rasio 80:20 ini didasarkan pada penggunaannya yang umum dan telah terbukti efektif dalam berbagai penelitian klasifikasi menggunakan *Random Forest* untuk memastikan model dilatih pada data yang cukup sambil tetap menyediakan porsi data yang memadai untuk pengujian [14][17] [41].

5. Pengecekan Distribusi Kelas (*Check Class Distribution*): Setelah pemisahan, dilakukan verifikasi untuk memastikan bahwa proporsi antara kelas 0 dan kelas 1 pada data latih dan data uji tetap representatif atau sebanding dengan distribusi pada *dataset* aslinya (total 757 data).

6. Rangkuman Data (*Summarize Data*): Tahap terakhir adalah merangkum jumlah data pada masing-masing set (latih dan uji) untuk memberikan gambaran yang jelas mengenai data yang akan digunakan pada tahap pemodelan, sebelum proses diakhiri.

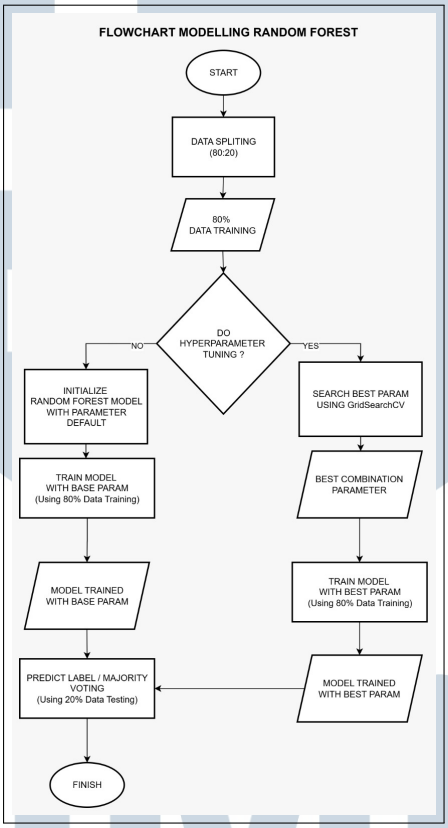
Dengan proses pembagian ini, dihasilkan dua set data yang siap digunakan: satu untuk melatih model dan satu lagi untuk menguji kemampuannya dalam melakukan generalisasi pada data baru.

3.9 Random Forest dan Hyperparameter Tuning

Setelah *dataset* siap digunakan, tahap selanjutnya adalah proses pemodelan dengan menggunakan algoritma *Random Forest*. Untuk mendapatkan pemahaman yang komprehensif, penelitian ini menerapkan dua pendekatan pemodelan. Pendekatan pertama adalah membangun model *baseline*, yang sengaja dibuat dengan memanfaatkan konfigurasi parameter standar atau *default* dari pustaka `scikit-learn` [16]. Tujuan dari pembuatan model *baseline* ini adalah untuk menetapkan sebuah titik acuan dasar (*benchmark*) yang merepresentasikan kinerja murni dari algoritma *Random Forest* sebelum dilakukan penyesuaian apapun.

Pendekatan kedua bertujuan untuk menemukan konfigurasi dengan performa terbaik. Pendekatan ini didasari oleh prinsip dalam *machine learning* bahwa setiap *dataset* memiliki karakteristik unik, sehingga konfigurasi parameter standar jarang menghasilkan performa yang maksimal [45]. Oleh karena itu,

dilakukan proses *hyperparameter tuning* yang melibatkan pencarian sistematis untuk menemukan kombinasi *hyperparameter* terbaik yang paling sesuai dengan pola data penelitian ini [39]. Pencarian ini memanfaatkan metode *GridSearchCV* untuk menguji berbagai konfigurasi parameter secara menyeluruh dan mendalam [40]. Alur kerja untuk membangun kedua versi model *baseline* dan performa terbaik diilustrasikan secara rinci pada Gambar 3.8.



Gambar 3.8. Flowchart Modelling Random Forest

Berdasarkan flowchart di atas, berikut adalah penjelasan rinci untuk setiap tahapan pemodelan dan optimasi yang dilakukan:

1. Persiapan Data Latih: Proses pemodelan dimulai dengan menggunakan 80% data latih yang telah dihasilkan dari tahap *data splitting*. *Dataset* ini menjadi input utama untuk melatih kedua skenario model yang akan dibandingkan.
2. Skenario 1: Pelatihan Model Baseline: Jalur pertama adalah membangun model baseline untuk menjadi tolok ukur performa.
    - *Inisialisasi Model*: Sebuah model *Random Forest* diinisialisasi menggunakan parameter *default* dari pustaka `scikit-learn` [16].

Parameter ini mencakup nilai-nilai standar seperti `n_estimators=100`, `max_depth=None`, dan `min_samples_split=2`.

- *Pelatihan Model*: Model dengan parameter dasar tersebut kemudian dilatih menggunakan 80% data latih. Hasil dari tahap ini adalah sebuah model terlatih dengan performa dasar (*Model Trained with Base Param*).

3. Skenario 2: Pelatihan Model dengan Optimasi: Jalur kedua bertujuan untuk menemukan dan melatih model dengan performa terbaik melalui *hyperparameter tuning*.

   - *Pencarian Parameter Terbaik*: Proses optimasi dilakukan dengan metode *GridSearchCV* untuk mencari kombinasi *hyperparameter* terbaik dari ruang pencarian yang telah ditentukan.
   - *Pelatihan Model Final*: Setelah *GridSearchCV* menemukan kombinasi parameter terbaik, sebuah model *Random Forest* baru dilatih ulang dari awal menggunakan parameter terbaik tersebut pada 80% data latih. Hasilnya adalah model terlatih versi optimasi (*Model Trained with Best Param*).

4. Konfigurasi Parameter untuk Tuning: Untuk membedakan kedua skenario, Tabel 3.2 menyajikan perbandingan antara parameter model *baseline* dan ruang pencarian (*grid*) yang digunakan dalam proses *hyperparameter tuning*.

Tabel 3.2. Perbandingan Set Parameter Model

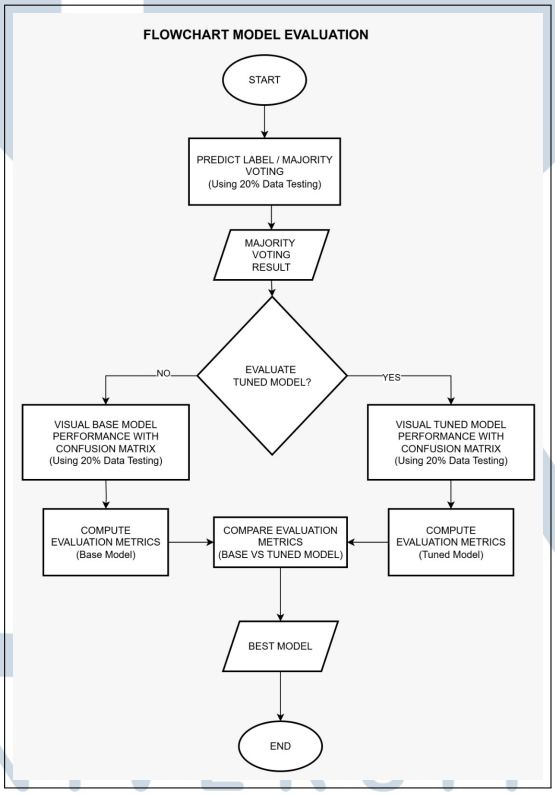
Parameter	**Nilai Model Baseline**	**Kombinasi Nilai pada Tuning**
`n_estimators`	100	[100, 200, 300]
`max_depth`	None	[10, 20, None]
`min_samples_split`	2	[2, 5, 10]
`min_samples_leaf`	1	[1, 2, 4]

Pemilihan rentang nilai parameter untuk *tuning* ini mengacu pada beberapa penelitian sebelumnya yang telah membuktikan efektivitasnya dalam meningkatkan akurasi model *Random Forest* [40][45]. Parameter lain yang tidak di-*tuning* seperti `criterion` dan `max_features` diatur ke nilai ‘’gini’‘ dan ‘’sqrt’‘ untuk kedua skenario.

5. Prediksi dan Evaluasi: Setelah kedua model (model *baseline* dan model hasil *tuning*) selesai dilatih, keduanya digunakan untuk membuat prediksi pada 20% data uji yang sama. Tahap ini penting untuk mengevaluasi dan membandingkan seberapa baik kinerja masing-masing model pada data yang belum pernah dilihat sebelumnya.

3.10 Evaluasi Model

Setelah model *baseline* dan model hasil *hyperparameter tuning* selesai dilatih, tahap krusial selanjutnya adalah mengevaluasi dan membandingkan kinerja keduanya. Tahapan evaluasi yang dilakukan diilustrasikan secara sistematis pada Gambar 3.9.



Gambar 3.9. Flowchart Evaluasi Model

Berdasarkan flowchart di atas, berikut adalah penjelasan rinci untuk setiap tahapan evaluasi model yang dilakukan:

1. Prediksi pada Data Uji: Langkah pertama adalah menggunakan kedua model yang telah dilatih (model *baseline* dan model hasil *tuning*) untuk membuat

prediksi label pada 20% data uji. Proses ini menghasilkan dua set hasil prediksi yang akan dievaluasi.

2. Visualisasi dengan *Confusion Matrix*: Untuk setiap model, dibuat sebuah *confusion matrix* guna memvisualisasikan distribusi hasil prediksi. Tabel ini menyajikan informasi mengenai jumlah prediksi yang benar dan salah, yang terbagi menjadi empat komponen utama: *True Positive* (TP), *False Positive* (FP), *True Negative* (TN), dan *False Negative* (FN) [14] [16].

3. Perhitungan Metrik Kinerja: Berdasarkan nilai dari *confusion matrix*, dilakukan perhitungan empat metrik evaluasi utama untuk mengukur kinerja setiap model secara kuantitatif. Metrik tersebut meliputi *Accuracy*, *Precision*, *Recall*, dan *F1-Score* [14].

4. Perbandingan dan Penentuan Model Terbaik: Tahap terakhir adalah membandingkan secara langsung metrik kinerja dari model *baseline* dengan model hasil *tuning*. Perbandingan ini bertujuan untuk menentukan pendekatan mana yang menghasilkan model terbaik (*Best Model*) dan menarik kesimpulan akhir mengenai efektivitas dari penerapan *hyperparameter tuning* dalam penelitian ini untuk menghasilkan performa terbaik dari algoritma *Random Forest*.