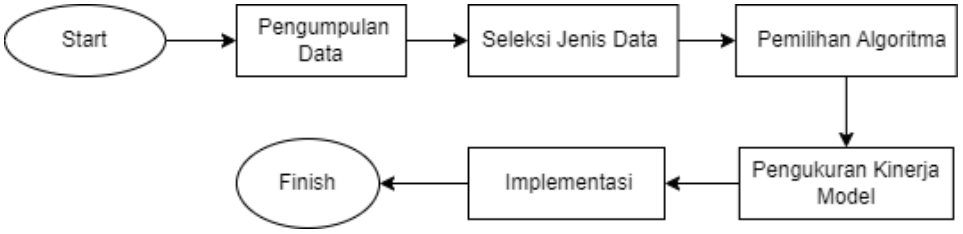
BAB III
METODOLOGI PENELITIAN

3.1. Metode Penelitian

Metode *K-Nearest Neighbor* (KNN) merupakan teknik untuk membuat model klasifikasi yang diterapkan dalam penelitian ini.

3.1.1 Model Requirements

Pada tahap kebutuhan model, dilakukan analisis dari pemilihan jenis data hingga penentuan teknologi atau platform yang digunakan untuk pembuatan model perancangan model dapat di terlihat pada gambar 3.1.



Gambar 3.1 Alur Model Requirements

Berdasarkan gambar 3.1 alur model requirements dapat di jelaskan sebagai berikut:

1. Pengumpulan Data

   Tahap pertama adalah pengumpulan data, dimana data diambil dari platform media sosial *twitter* yang bersifat publik. Data yang dikumpulkan berupa teks atau tipe data string, dengan rentang waktu pengumpulan dari 20 Oktober 2023 – 31 Januari 2024.

2. Seleksi Jenis Data

   Selanjutnya, data yang telah terkumpul akan melalui tahap seleksi jenis data. Pada tahap ini, data diverifikasi untuk memastikan validitasnya, dan dipilih serta disaring data yang relevan sesuai dengan tujuan penelitian.

3. Pemilihan Algoritma

   Setelah itu, dilakukan pemilihan algoritma untuk mengidentifikasi *cyberbullying*, dimana dalam penelitian ini menggunakan algoritma *K-Nearest Neighbor* (KNN).

4. Pengukuran Kinerja Model

   Setelah itu, dilakukan pemilihan algoritma untuk mengidentifikasi *cyberbullying*, dimana dalam penelitian ini menggunakan algoritma *K-Nearest Neighbor* (KNN).

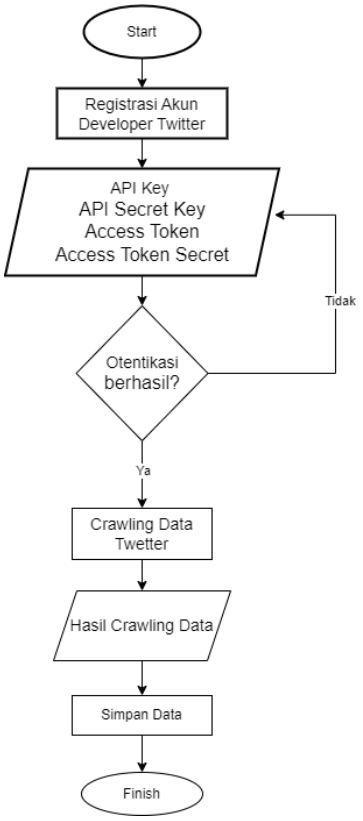
5. Implementasi

   Setelah itu, dilakukan pemilihan algoritma untuk mengidentifikasi cyberbullying, dimana dalam penelitian ini menggunakan algoritma *K-Nearest Neighbor* (KNN).

3.1.2 Data Collection

Dalam penelitian ini, tahap *crawling* data dilakukan pada layanan microblogging *twitter* dengan fokus pada *tweet* dalam bahasa Indonesia. Untuk menjalankan proses *crawling* data pada *twitter*, pengguna harus memiliki akun *twitter* pribadi dan mengakses *twitter API*.

Pengumpulan data dari *twitter* dilakukandengan mengambil data secara langsung *(crawling data)* menggunakan *google colab*. Proses ini dijelaskan dalam gambar 3.2 berikut:



Gambar 3.2 Alur *Crawling*

Berdasarkan gambar 3.2 di atas, proses pengambilan data dapat dijelaskan sebagaiberikut:

1. *Registrasi* Akun *Developer Twitter*

   Mendaftarkan akun sebagai pengembang *twitter* di situs *web* *https://developer.twitter.com/en* untuk mendapatkan izin penggunaan *twitter API.*

2. *Autentikasi* ke *Twitter API*

   Setelah berhasil mendaftar sebagai pengembang *twitter* di situs tersebut, penulis memperoleh *Twitter API* yang terdiri dari empat koderahasia: *consumer key, consumer secret, access token*, dan *access secret*

3. *Crawling* Data

   Setelah proses otentikasi berhasil, pengguna dapat menggunakan *twitter API* untuk menarik data *tweet*. Namun, pengguna standar terbatas hanya dapat mengambil data hingga 1 minggu ke belakang, dengan batasan maksimal 500.000 *tweets* per bulan.

4. Hasil *Crawling* Data

   Data *tweet* yang diambil terbatas pada *tweet* yang mengandung kata kunci yang telah di tentukan dapat dilihat pada tabel 3.1. yang sudah termasuk dengan kata kunci tanggal bulan dan tahun yang digunakan dalam penelitian ini.

Tabel 3.1 Kata Kunci

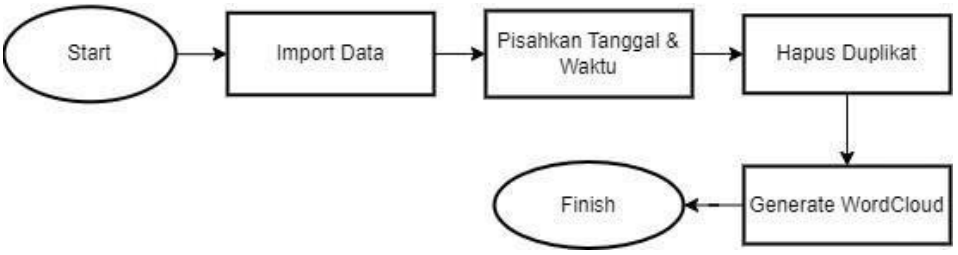
No	Kata Kunci	Tanggal
1	Gibran	20 Oktober – 31 Oktober 2023
2	Debat	31 Desember 2023 -17 Januari 2024
3	Mundur	31 Januari 2024

5. Simpan Data

   Setelah mendapatkan data, langkah selanjutnya adalah menyimpan data tersebut, data disimpan dalam format *csv*.

3.1.3 Prepocessing Data

Proses *preprocessing* data merupakan langkah penting dalam analisis data untuk memastikan kualitas dan kebersihan data sebelum analisis lebih lanjut. Berikut adalah langkah-langkah yang dilakukan dalam *preprocessing* data dapat dilihat pada gambar 3.3



Gambar 3.3 Alur *Prepocessing Data*

Berikut adalah penjelasan dari gambar 3.3 di atas :

1. *Import* Data

   Langkah pertama adalah mengimpor *library pandas* yang digunakan untuk manipulasi dan analisis data. Data dibaca dari file CSV menggunakan *pd.read_csv.*

2. Memisahkan Kolom Tanggal dan Waktu

   Setelah data diimpor, kolom *'created_at*' yang berisi informasi tanggal dan waktu dipisahkan menjadi dua kolom terpisah, yaitu 'Tanggal' dan 'Waktu'. Ini dilakukan dengan mengubah tipe data kolom *'created_at'* menjadi tipe *datetime*, lalu mengekstrak bagian tanggal dan waktu secara terpisah.

3. Proses Hapus Data Duplikat

   Data duplikat bisa menyebabkan analisis menjadi tidak akurat karena menghitung data yang sama lebih dari sekali.

4. *WordCloud* Sebelum Preprocessing

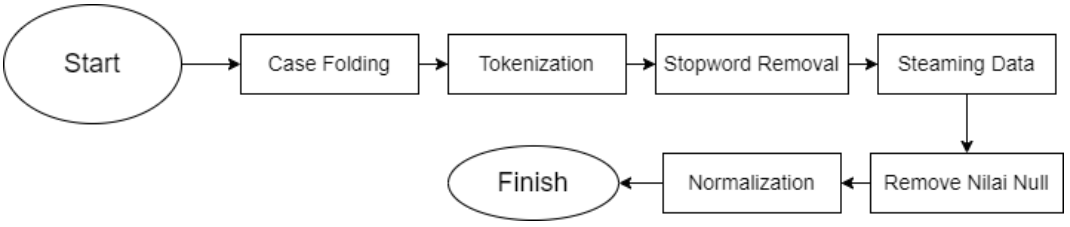
   Sebelum melakukan *preprocessing* lebih lanjut seperti pembersihan teks, tokenisasi, dan lainnya, dibuat *WordCloud* untuk visualisasi kata-kata yang paling sering muncul dalam data. Ini memberikan gambaran awal tentang konten teks.

3.1.4 Data Cleaning

Proses ini bertujuan untuk menyusun data menjadi siap dan layak untuk dianalisis. Pada tahapan ini terdiri dari dua bagian: tahap pra-proses sebelum data dibagi menjadi data training dan data testing, serta pra-proses pada tahap pemrosesan dokumen. Pembersihan yang dilakukan pada tahap pra-proses termasuk penghapusan *URL, HTML, Emoji,, Simbol, number, username.*

Dalam tahap selanjutnya melibatkan penggunaan operator "*Case Folding*, "*Tokenize*", "*Stopword Removal*", "*Steaming Data*", "*Remove Nilai Nul*" dan "Normalization" yang menggunakan kamus dalam bahasa Indonesia untuk mendeteksi perkata dan perkalimat. Ini diilustrasikan dalam Gambar 3.3 berikut:

Berikut adalah penjelasan dari gambar 3.4 di atas:



Gambar 3.4* Alur *Cleaning

1. *Case Folding*

   Langkah pertama adalah mengubah semua karakter teks menjadi huruf kecil. Hal ini dilakukan untuk menghindari duplikasi kata yang mungkin terjadi akibat perbedaan kapitalisasi.

2. *Tokenization*

   Tahap ini memecah teks menjadi unit-unit yang lebih kecil, yang disebut token. Token bisa berupa kata, frasa, atau entitas lain seperti angka atau tanda baca. Proses tokenisasi ini mempermudah analisis teks dengan memisahkan teks menjadi elemen-elemen yang dapat diolah lebih lanjut.

3. *Stopword Removal*

   Proses penghapusan *stopword* dan penyaringan dilakukan karena sebagian besar data teks mengandung kata-kata umum yang tidak memiliki makna penting. Hal ini dapat memengaruhi akurasi analisis. Biasanya, kata-kata ini muncul dalam frekuensi yang cukup tinggi. Contoh *stopword* dalam Bahasa Inggris termasuk kata-kata seperti "*are*", "*is*", "*i*", "*am*", "*was*", "were", "they", "you", "the", dan lain-lain.Proses

penyaringan juga melibatkan penambahan manual daftar kata stoplist tambahan yang dianggap tidak bermakna.

4. *Steaming Data*

   Dalam konteks pemilu, proses stemming dapat mengubah kata-kata seperti "memilih" menjadi "pilih" atau "kandidat" menjadi "kandidat".

5. *Remove Nilai Null*

   Jika ada nilai-nilai null atau kosong dalam dataset teks, langkah ini melibatkan penghapusan atau penanganan nilai-nilai tersebut agar data tetap konsisten dan bersih sebelum dilanjutkan ke tahap analisis lebih lanjut.

6. *Normalization*

   Normalisasi mengacu pada proses mengubah atau mengembalikan teks ke bentuk standar atau normal. Ini bisa termasuk penggantian sinonim atau variasi kata dengan bentuk standar, misalnya mengubah kata-kata non-formal seperti "gimana" menjadi bentuk standar "bagaimana”.

3.1.6 Data Labeling

Dalam penelitian ini, data akan dilabeli melalui *Natural Language Processing* yang ada dalam *Library Python* yaitu *Textblob* Setelah data diperoleh dan disimpan dalam file CSV, langkah selanjutnya adalah melakukan pelabelan *tweet* secara manual. *Tweet* akan diberi label sesuai dengan sentimennya, dengan dua kategori yang ditetapkan oleh penulis, yaitu "ya"dan "tidak".

3.1.7 Feature Engineering

Data *training* terdiri dari 70% dari total data, sementara data *testing* terdiri dari 30% sisanya. Pada tahap *Feature Engineering*, metode engineeringyang diterapkan yaitu TF-IDF.

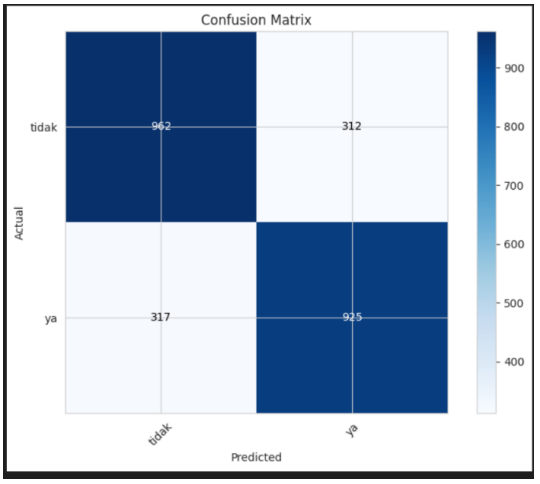
3.1.8 Model Training

Proses pelatihan model merupakan tahapan di mana model *machine learning* dilatih menggunakan algoritma *K-Nearest Neighbor* dan *Support Vektor Machine*. Setelah data tweet diubah menjadi vektor, langkah selanjutnya adalah membagi dataset menjadi data latih dan data uji menggunakan fungsi *train-test-split* pada *library Python*. Model akan dilatih menggunakan tiga

skenario pembagian komposisi data yang berbeda.

3.1.9 KNN Confusion Matrix

Berikut adalah hasil dari confusion matrix yang dihasilkan oleh model klasifikasi *Machine Learning*:



Gambar *3.5 KNN Confusion Matrix* Perkalimat

True Positives (TP) = 962

True Negative (TN) = 925

False Positives (FP) = 312

False Negatives (FN) = 317

Cofusion Matrix pada gambar 3.5 memperlihatkan 962 + 925 = 1887 prediksi benar dan 312 +317 = 629 prediksi salah. Setelah mendapatkan nilai *confusion matrix*, langkahselanjutnya adalah menghitung hasil dari poin-poin di bawah ini:

1. *Accuracy*

$$Accuracy = \frac{962 + 925}{(962+925+312+317)} = 0{,}75 \quad (3.1)$$

Matriks ini dihitung dengan membagi jumlah prediksi yang benar (*True Positives dan True Negatives*) dengan jumlah total sampel pada data pengujian. Ini memberikan gambaran tentang seberapa baik model dapat memprediksi kelas data yang benar, dengan memperhitungkan baik prediksi positif maupun negatif.

Nilai akurasi sebesar 0.75 (atau 75%) menunjukkan bahwa model memiliki tingkat keakuratan sebesar 75% dalam memprediksi kelas data untuk memprediksi perkalimat. Dengan kata lain, dari keseluruhan data yang diprediksi, 75% di antaranya diprediksi dengan benar oleh model.

2. *Precision*

$$Precision = \frac{962}{(962+312)} = 0{,}755102041 \quad (3.2)$$

Precision memberikan gambaran tentang seberapa akurat model dalam mengidentifikasi contoh yang sebenarnya positif. Nilai *precision* sebesar 0,75 berarti sebanyak 75% dari prediksi positif yang dilakukan oleh model adalah benar-benar positif dan relevan.

3. *Recall*

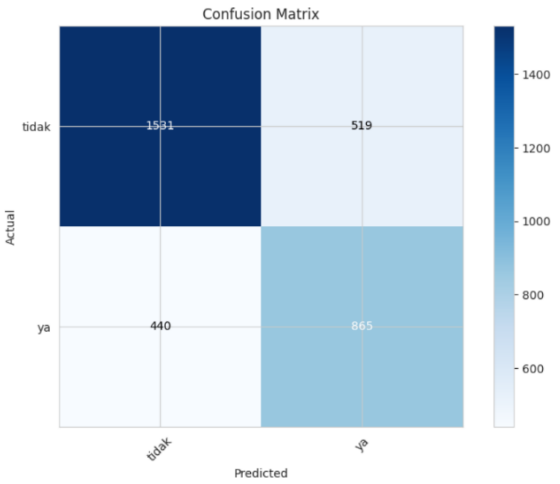
$$Recall = \frac{962}{(962+317)} = 0{,}752150117 \quad (3.3)$$

Recall membantu dalam mengevaluasi seberapa baik model dapat mengidentifikasi semua contoh dari kelas yang positif. Dengan nilai *Recall* sebesar 0,75, ini berarti model mampu mengidentifikasi 75% dari semua kasus positif yang ada di *dataset*.

4. *F1-Score*

$$F1 - Score = \frac{2\times(0{,}755102041\times0{,}752150117)}{(0{,}755102041+0{,}752150)} = 0{,}753623189 \quad (3.4)$$

F1-score dihitung dengan mengambil nilai *harmonic mean*dari *Precision* dan *Recall*. *Harmonic mean* digunakan di sini karenamemperhitungkan kedua metrik secara seimbang, tidak seperti *mean* aritmatika yang bisa didominasi oleh satu metrik jika nilainya sangat besar. *F1-score* sebesar 0,753623189 menunjukkan seberapa baik model dapat mencapai keseimbangan antara *Precision* dan *Recall.* Semakin tinggi nilai *F1-score*, semakin baik keseimbangan antara *Precision* dan *Recall* model tersebut.



Gambar *3.6 KNN Confusion Matrix* Perkata

True Positives (TP) = 1531

True Negative (TN) = 865

False Positives (FP) = 519

False Negatives (FN) = 440

Cofusion Matrix pada gambar 3.6 memperlihatkan 1531 + 865 = 2396 prediksi benar dan 519 + 440 = 959 prediksi salah. Setelah mendapatkan nilai *confusion matrix*, langkah selanjutnya adalah menghitung hasil dari poin-poin di bawah ini:

1. *Accuracy*

$$Accuracy = \frac{1531+865}{(1531+865+519+440)} = 0{,}714157973 \quad (3.5)$$

Hasil prediksi akurasi pada predeksi jumlah perkata menghasilkan hasil yang lebuh rendah dibandingan dengan deteksi perkalimat, model pada kasus ini memiliki performa yang sedikit lebih rendah dalam memprediksi secara benar.

2. *Precision*

$$Precision = \frac{1531}{(1531+519)} = 0{,}746829268 \quad (3.6)$$

Hasil *precision* untuk model pendeteksi perkata pada kasus ini mendapatkan hasil sebesar 0,7468 atau sama dengan 74%.

3. *Recall*

$$Recall = \frac{1531}{(1531+440)} = 0{,}776763064 \quad (3.7)$$

Hasil *recall* pada model pendeteksi berdasarkan perkatamemperlihatkan peningkatan dari sebelumnya yaitu 0,7767 atau sama dengan 77%.

4. *F1-Score*

$$F1 - Score = \frac{2\times(0{,}746829268\times 0{,}776763064)}{((0{,}746829268+0{,}776763064))} = 0{,}761502114 \quad (3.8)$$

Hasil predeksi model *F1-score* mengalami peningkatan nilai *F1-Score* menunjukkan sedikit peningkatan dalam keseimbangan antara *precision* dan *recall*, yang berarti model mungkin sedikit lebih baik dalam memprediksi kelas positif dan negatif secara seimbang.

3.1.10 Model Evaluation

Setelah pemodelan KNN selesai, evaluasi dilakukan menggunakan confusion matrix untuk mengukur kinerja model dalam memprediksi data uji. Hasil evaluasi ini mencakup beberapa parameter, termasuk akurasi (accuracy), presisi (precision), recall, dan F1-score (Dalvi dkk., 2020).

3.1.11 Operation

Operation dalam *Machine learning* menggunakan KNN untuk mendeteksi kasus *cyberbullying* di media sosial *Twitter* melibatkan *deployment model*, *monitoring*, dan *updating* untuk meningkatkan akurasinya. Model tersebut harus dipantau secara *real-time* untuk memastikan model tersebut mendeteksi *cyberbullying* secara akurat. *Updating* terhadap model ini juga penting untuk memastikan model ini tetap efektif ketika jenis *cyberbullying* baru bermunculan *(Raj dkk., 2022).*