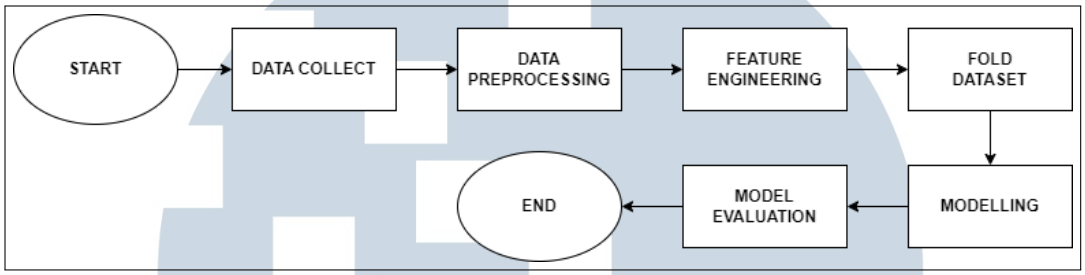
BAB 3
METODOLOGI PENELITIAN



Gambar 3.1. Metodologi Penelitian

Penelitian ini dilakukan melalui tahapan-tahapan sebagaimana ditunjukkan pada Gambar 3.1. Proses ini bertujuan untuk menghasilkan model yang mampu mendeteksi kesalahan dalam penggunaan tanda baca.

3.1 *Data Collection*

Tahapan pengumpulan data (*data collection*) terdiri dari dua bagian utama, yaitu pengumpulan berita sebagai *dataset* utama dan pengumpulan gelar akademik sebagai *dataset* pendukung. Kedua jenis *dataset* dikumpulkan dengan metode yang berbeda.

3.1.1 *Dataset* Utama

Dataset utama berupa kumpulan berita yang diberikan secara langsung oleh dosen penelitian melalui layanan OneDrive. Seluruh data memiliki format `*.eml`. Data tersebut diolah secara bertahap dengan mengonversi format `*.eml` menjadi `*.pdf`, kemudian ke `*.txt`, dan akhirnya ke `*.csv`. Setelah data diubah menjadi format `*.csv`, seluruh berita digabung menjadi satu *file* `*.csv` baru. Selanjutnya, berita-berita tersebut dipecah menjadi kalimat-kalimat dengan syarat bahwa setiap kalimat harus diakhiri dengan tanda titik (.), tanda tanya (?), atau tanda seru (!).

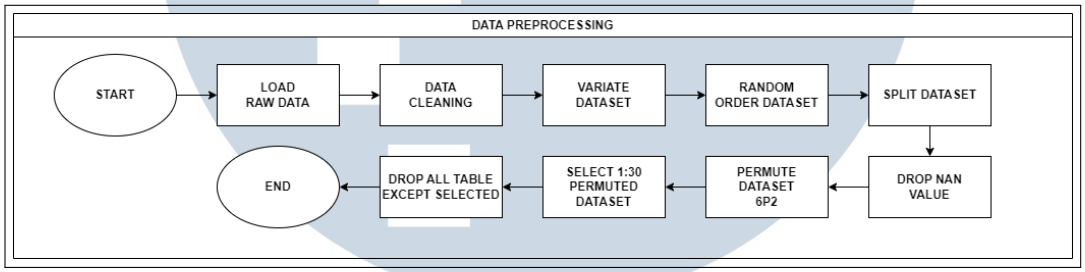
3.1.2 *Dataset* Pendukung

Dataset pendukung berupa daftar gelar akademik [19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36] digunakan untuk pengecualian

kata-kata dalam kalimat yang tidak boleh dihapus atau terpotong selama proses pengolahan data, karena mengandung tanda baca yang tidak menjadi fokus penelitian. Pengumpulan *dataset* ini dilakukan dengan dua cara:

1. ***Scraping***: Metode ini digunakan jika *dataset* berbentuk tabel. Data diambil langsung dari situs web menggunakan teknik *scraping*.
2. ***Manual Input***: Metode ini diterapkan jika *dataset* berbentuk tulisan naratif dan memerlukan tambahan informasi yang harus dicari di situs lain.

3.2 Data Preprocessing



Gambar 3.2. Urutan *Data Preprocessing*

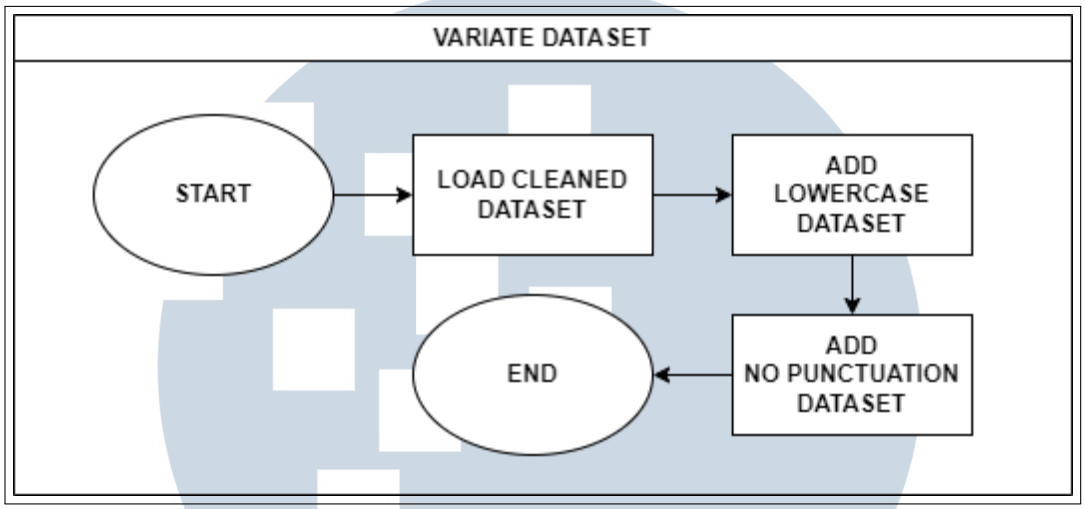
3.3 *Data Preprocessing*

Dataset yang digunakan untuk kebutuhan pemodelan sering kali ditemukan dalam kondisi yang kurang terstruktur. Oleh karena itu, *dataset* utama yang telah dikumpulkan perlu melalui tahapan pemrosesan agar pengaplikasian fitur dapat dilakukan secara optimal, seperti yang ditunjukkan pada Gambar 3.2.

3.3.1 Data Cleaning

Pada tahap ini, dilakukan pembersihan (*data cleaning*) terhadap bagian-bagian dari *dataset* yang tidak relevan untuk diolah. Bagian *dataset* yang tidak relevan mencakup *news header* dan *news footer*, yaitu informasi tambahan di luar isi berita utama.

3.3.2 *Variate Dataset*



Gambar 3.3. Urutan Variasi Data

3.4 *Data Variation*

Proses variasi data dilakukan seperti yang ditunjukkan pada Gambar 3.4 untuk membantu model memahami berbagai bentuk kalimat (penjelasan lebih lanjut akan diberikan setelah data dipermutasi). Bentuk variasi dataset yang digunakan untuk permutasi antara lain:

1. *dataset* utama dalam bentuk orisinal,
2. *dataset* utama yang diubah menjadi huruf kecil (*lowercase*), dan
3. *dataset* utama yang dihilangkan tanda bacanya (*no punctuation*).

3.4.1 *Random Order Dataset*

Pengacakan urutan dataset dilakukan untuk membantu permutasi menghasilkan paragraf dengan kalimat yang lebih variatif. Urutan dari masing-masing kolom (*Kalimat*, *lowercase*, dan *no punctuation*) dibuat secara acak sehingga setiap baris yang dihasilkan memiliki kombinasi kalimat yang berbeda.

3.4.2 *Split Dataset*

Dataset yang telah diacak urutannya kemudian dibagi menjadi dua bagian, sehingga menghasilkan enam kolom: *Kalimat I*, *Kalimat II*, *lowercase I*, *lowercase*

II, *no punctuation I*, dan *no punctuation II*. Pembagian ini bertujuan agar dataset yang akan diolah dapat menghasilkan hasil yang lebih merata dan mencegah terjadinya *data imbalance*.

3.4.3 *Permute Dataset*

Jika dataset *Kalimat*, *lowercase*, dan *no punctuation* digabung menjadi satu, seluruh dataset yang dihasilkan akan bersifat *invalid*. Hal ini terjadi karena data *lowercase* dan *no punctuation* sengaja dibentuk untuk membantu model mengenali kesalahan penggunaan tanda baca. Oleh karena itu, dilakukan permutasi agar dataset mengandung data valid (contoh: *Kalimat I* dan *Kalimat II*) serta data invalid (contoh: *Kalimat I* dan *no punctuation II*). Permutasi yang digunakan adalah $6P2$, yang dihitung sebagai berikut:

$$n = 6, \quad r = 2$$

$$nPr = \frac{n!}{(n-r)!}$$

$$6P2 = \frac{6!}{(6-2)!} = \frac{6!}{4!} = \frac{720}{24} = 30$$

Total variasi permutasi yang didapatkan adalah 30 kemungkinan.

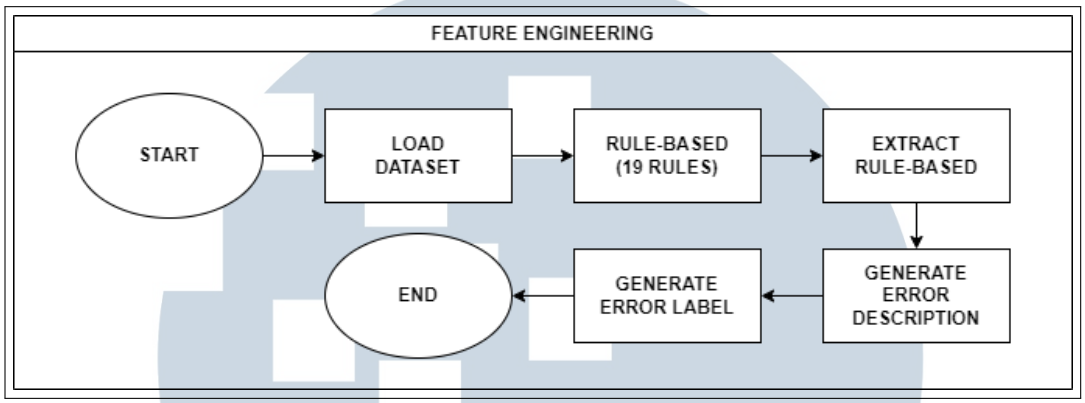
3.4.4 *Select Data*

Dari 30 kolom yang berisi berbagai kemungkinan permutasi, dibuat satu kolom baru bernama *selected*. Kolom ini digunakan untuk memilih salah satu permutasi secara acak dari 30 kemungkinan pada setiap baris. Pemilihan secara acak ini bertujuan untuk menghasilkan dataset yang lebih bervariasi sekaligus mengurangi beban komputasi.

3.4.5 *Drop Columns*

Pembersihan kolom dilakukan dengan hanya menyisakan kolom *selected*. Kolom *selected* menjadi hasil akhir dari proses *preprocessing*, yang kemudian akan digunakan untuk pendeteksian kesalahan penggunaan tanda baca.

3.5 *Feature Engieering*



Gambar 3.4. Urutan *Feature Engineering*

Pada tahap ini, data yang telah lolos pada tahap *preprocessing* akan diolah sehingga dapat meningkatkan kinerja dan efisiensi model RNN yang nantinya akan digunakan untuk melatih dan memprediksi kesalahan dari penggunaan tanda baca seperti yang ditunjukkan pada Gambar 3.5.

3.5.1 *Rule-Based*

Pada tahap ini, dibentuklah 19 aturan untuk mendeteksi penggunaan tanda baca. Aturan yang dibentuk telah disesuaikan dengan EYD V dan kebutuhan jurnalis untuk menyunting naskah. Adapun aturan yang telah dibuat adalah sebagai berikut:

1. penggunaan tanda titik, tanya, dan seru di akhir kalimat,
2. penggunaan tanda tanya untuk kalimat tanya (mengandung kata tanya seperti siapa, apa, di mana, mengapa, kapan, dan bagaimana),
3. penggunaan huruf kecil apabila kata sebelumnya tidak diakhiri oleh tanda titik, tanya, dan seru,
4. penggunaan tanda titik, tanya, dan seru sebelum huruf kapital pada kalimat berikutnya,
5. penggunaan tanda seru pada sebuah ekspresi,
6. penggunaan tanda titik untuk format waktu (misal: 18.00 WIB),

7. tidak menggunakan titik sebagai pemisah angka untuk format tanggal, bulan, dan waktu,
8. penggunaan titik sebagai pemisah ribuan untuk kuantitas,
9. tidak menggunakan titik sebagai pemisah ribuan pada non-kuantitas,
10. penggunaan tanda koma sebelum kata hubung dan atau atau,
11. penggunaan tanda koma sebelum kata hubung kalau, karena, atau agar,
12. penggunaan tanda koma sebelum tanda petik (Kecuali kalimat langsung berada di awal kalimat atau sebelum tanda petik sudah ada tanda titik, tanya, atau seru),
13. penggunaan tanda koma setelah kata hubung tetapi, melainkan, atau sedangkan,
14. penggunaan tanda koma setelah kata hubung oleh karena itu, jadi, atau meskipun demikian,
15. penggunaan tanda koma setelah kata o, wah, nak, bu, atau dik,
16. penggunaan tanda koma setelah kata yang tertulis setelah Jl. atau jalan,
17. penggunaan tanda koma setelah kata dengan hormat, salam sejahtera, atau hormat kami,
18. penggunaan tanda koma sebelum dan atau setelah kata gelar, dan
19. penggunaan tanda koma setelah kata dalam atau atas.

3.5.2 *Extract Rule-Based*

Ekstrasi dari aturan berfungsi untuk mengimplementasikan aturan-aturan yang dapat digunakan untuk mengenali pola, membuat keputusan, atau melakukan klasifikasi berdasarkan logika yang telah dibentuk. Hal ini dapat mengidentifikasi hubungan atau pola yang jelas dalam data.

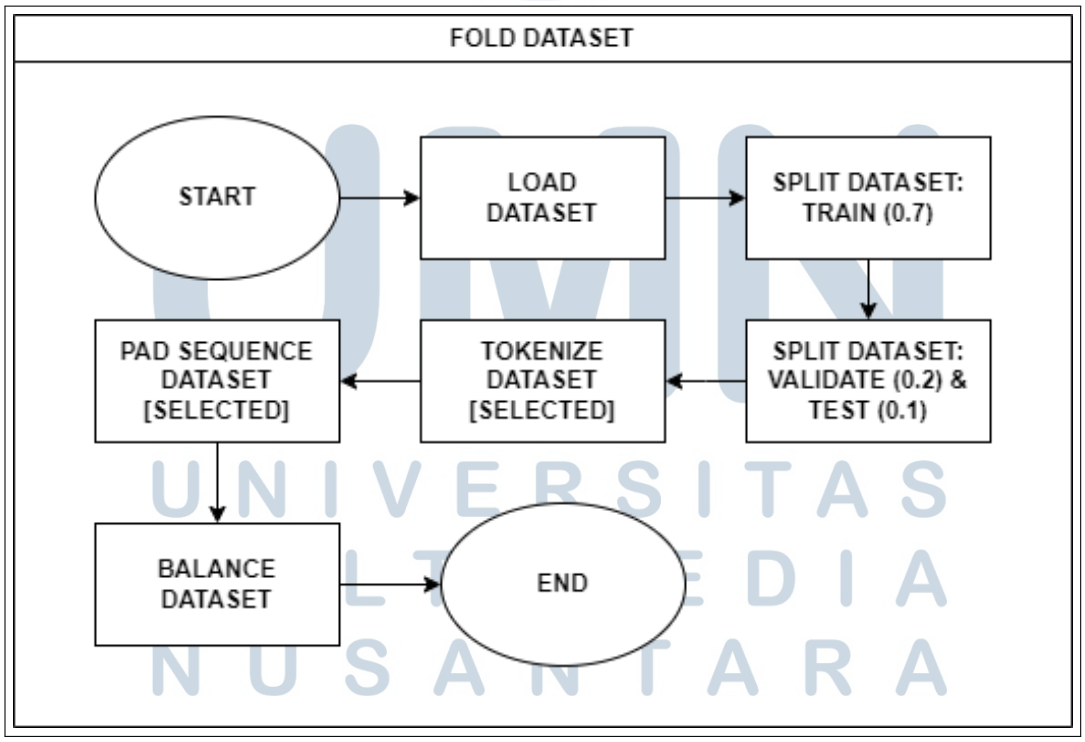
3.5.3 *Generate Error Description*

Langkah ini dilakukan untuk menghasilkan penjelasan dari kesalahan pada penggunaan tanda baca pada dataset yang lolos tahap *preprocessing*. Apabila terdapat lebih dari satu kesalahan penggunaan tanda baca, maka seluruh penjelasan akan digabung menjadi satu. Apabila penggunaan tanda baca sudah benar, maka deskripsi dari kesalahan penggunaan tanda baca akan menghasilkan penjelasan kosong (*NaN Value*).

3.5.4 *Generate Error Label*

Langkah ini berfungsi untuk membuat label untuk setiap data dalam dataset guna menunjukkan apakah data tersebut mengandung kesalahan (invalid, dengan label 0) atau tidak (valid, dengan label 1). Label ini nantinya digunakan sebagai variabel target (y) dalam proses pelatihan model RNN.

3.6 *Fold Dataset*



Gambar 3.5. Urutan *Feature Engineering*

Tahap *fold dataset* yang digunakan sesuai pada Gambar 3.6 berfungsi untuk membantu mengevaluasi kinerja model RNN. Hal ini dapat membuat konsistensi pada kinerja dari model RNN dan menghindari model dari *overfitting* dan bias terhadap suatu target.

3.6.1 *Split Dataset*

Langkah ini adalah membagi dataset sesuai dengan porsi tertentu agar dataset dapat melakukan pelatihan, memvalidasi, dan memprediksi kinerja dari model secara berurutan. Pada langkah ini dataset dibagi menjadi tiga subset:

1. subset pelatihan sebesar 70%,
2. subset validasi sebesar 20%, dan
3. subset prediksi sebesar 10%.

3.6.2 *Tokenization*

Kolom *selected* pada dataset utama yang berbentuk *string* dilakukan tokenisasi agar dapat diubah menjadi numerik. Hal ini dapat membantu model lebih mudah memahami pola dalam sebuah data yang diberikan.

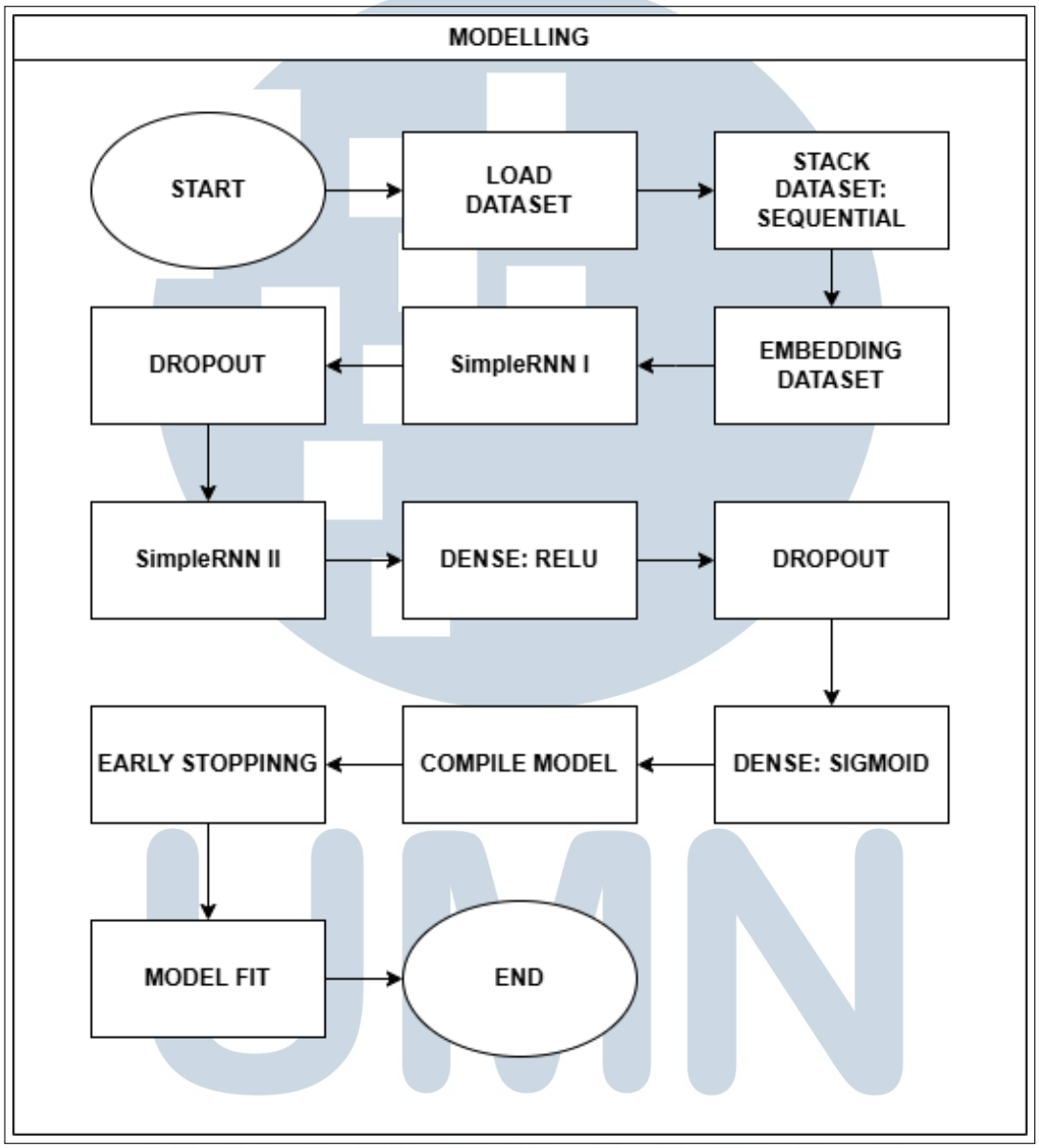
3.6.3 *Pad Sequence*

Setelah dilakukan tokenisasi, kolom *selected* yang telah berubah menjadi token kemudian ditambahkan nilai 0 ke dalam urutan token. Hal ini dapat membantu dalam pembentukan panjang token yang seragam sehingga model dapat diproses secara *batch*.

3.6.4 *Balance Dataset*

Pada tahap ini dilakukan penyeimbangan dataset yang disebabkan oleh ketidakseimbangan data dengan label valid (minoritas) dan label invalid (mayoritas). Penyeimbangan data dilakukan agar model tidak bias dalam memprediksi penggunaan tanda baca yang salah maupun benar.

3.7 *Modelling*



Gambar 3.6. Urutan *Modelling*

Proses *modelling* dilakukan melalui beberapa langkah utama seperti berikut:

1. **Load Dataset:** Dataset dimuat ke dalam sistem untuk persiapan proses pelatihan.
2. **Stack Dataset: Sequential:** Dataset disusun dalam bentuk berurutan agar sesuai dengan arsitektur *Sequential Model*.

3. **Embedding Dataset:** Data teks direpresentasikan sebagai vektor numerik menggunakan lapisan embedding untuk menangkap hubungan semantik antar kata.

4. **SimpleRNN I dan II:** Dua lapisan *Simple Recurrent Neural Network (SimpleRNN)* diterapkan secara berurutan untuk memahami pola sekuensial dalam data.

5. **Dropout:** Lapisan *dropout* disisipkan untuk mencegah *overfitting* dengan menonaktifkan sebagian node selama pelatihan.

6. **Dense Layers:** Lapisan *dense* dengan fungsi aktivasi *ReLU* dan *sigmoid* diterapkan untuk transformasi data non-linear dan prediksi akhir.

7. **Compile Model:** Model dikompilasi dengan menetapkan fungsi *loss*, optimisasi, dan metrik evaluasi.

8. **Early Stopping:** Strategi *early stopping* digunakan untuk menghentikan pelatihan saat performa model tidak lagi meningkat.

9. **Model Fit:** Model dilatih dengan data yang telah diproses dan siap untuk evaluasi.

3.8 Evaluation

Proses evaluasi model dilakukan melalui beberapa langkah penting untuk menilai performa model, yang dijelaskan sebagai berikut:

1. **ROC Curve dan Optimal Threshold:**
   - Kurva *Receiver Operating Characteristic (ROC)* digunakan untuk mengevaluasi kemampuan model dalam membedakan kelas positif dan negatif.
   - Probabilitas hasil prediksi pada data validasi (`y_val_probs`) dihitung terlebih dahulu dengan fungsi `model.predict`.
   - Fungsi `roc_curve` dari `sklearn.metrics` menghasilkan tiga nilai:
     - `fpr` (False Positive Rate)
     - `tpr` (True Positive Rate)
     - `thresholds` (Ambang batas untuk memisahkan kelas)

- Ambang batas optimal (*optimal threshold*) ditentukan dengan memaksimalkan nilai $tpr - fpr$ menggunakan indeks optimal:

```
optimal_threshold = thresholds[np.argmax(tpr - fpr)]
```

2. **Evaluasi pada Data Uji:**

   - Data uji (`X_test`) diprediksi menggunakan ambang batas optimal:

```
y_pred_test = (model.predict(X_test) >= optimal_threshold).astype(int)
```

   - Hasil prediksi dibandingkan dengan label sebenarnya (`y_test`) menggunakan laporan klasifikasi dari fungsi `classification_report`.
   - Laporan klasifikasi mencakup metrik berikut:
     - **Precision:** Ketepatan prediksi positif terhadap semua prediksi positif.
     - **Recall:** Kemampuan model dalam mendeteksi kelas positif.
     - **F1-Score:** Rata-rata harmonis antara *precision* dan *recall*.
     - **Support:** Jumlah sampel untuk setiap kelas.