BAB 3
METODOLOGI PENELITIAN

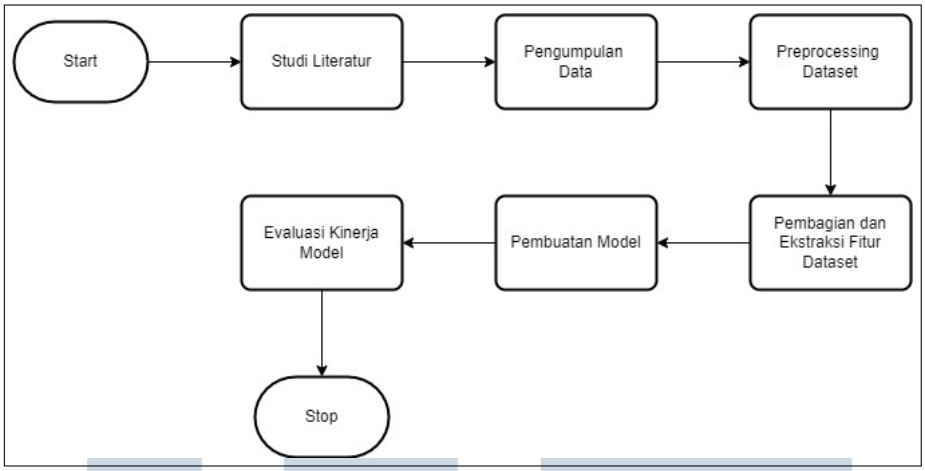
Penelitian ini dilakukan menggunakan beberapa spesifikasi perangkat keras dan perangkat lunak yang dirancang untuk mendukung proses analisis data dan pembuatan model klasifikasi. Tabel 3.1 merangkum spesifikasi utama yang digunakan selama penelitian.

Tabel 3.1. Spesifikasi Perangkat Keras dan Perangkat Lunak Penelitian

Kategori	**Spesifikasi**
Perangkat Keras	
CPU	AMD Ryzen 5 5600
GPU	NVIDIA RTX 2060 Super
RAM	32GB DDR4 3200MHz (2x16GB)
Penyimpanan	SSD 512GB + SSD 1TB
Perangkat Lunak	
OS	Windows 11 Pro 64-bit
Bahasa Pemrograman	Python 3.10.16 (Conda Environment)
Library Utama	Scikit-learn, LightGBM, Numpy, Pandas, Matplotlib
IDE	JetBrains DataSpell
Framework Pendukung	Optuna (Optimasi Parameter)

3.1 Metodologi Penelitian

Gambar 3.1 memperlihatkan metode dan tahapan yang diterapkan pada penelitian.



Gambar 3.1. *Flowchart Penelitian*

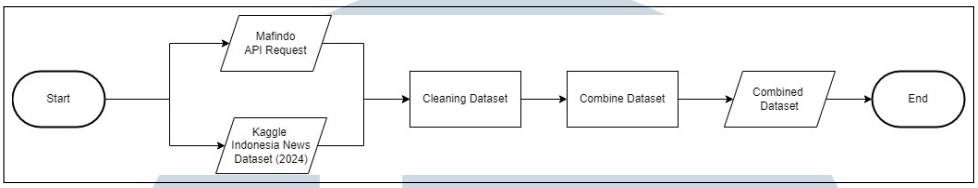
3.2 Studi Literatur

Tahap studi literatur dilakukan dengan menelusuri berbagai penelitian yang berkaitan dengan klasifikasi berita hoaks, khususnya dalam konteks Indonesia. Penelitian-penelitian sebelumnya umumnya menggunakan algoritma *machine learning* untuk menyelesaikan tugas klasifikasi, dengan teknik berbasis *ensemble learning* seperti *Random Forest*, *Gradient Boosting*, dan *XGBoost* yang sering menjadi pilihan karena kemampuannya dalam menangani data dengan karakteristik kompleks.

Selain itu, banyak penelitian juga mengeksplorasi metode representasi teks untuk mengekstraksi fitur sebelum diterapkan ke algoritma *machine learning*. Teknik-teknik yang umum digunakan mencakup *Term Frequency-Inverse Document Frequency* (TF-IDF), *CountVectorizer* yang bertujuan untuk menangkap aspek semantik dan sintaktik dari teks secara efektif.

Berbagai pendekatan yang digunakan dalam studi sebelumnya menunjukkan bahwa penggabungan teknik representasi teks dengan algoritma *machine learning* dapat meningkatkan akurasi model dalam klasifikasi berita hoaks. Temuan-temuan tersebut menjadi dasar bagi penelitian ini untuk menyusun strategi klasifikasi yang relevan dalam konteks bahasa Indonesia.

3.3 Pengumpulan Data



Gambar 3.2. *Pengumpulan Data*

Data berita hoaks diperoleh melalui *Application Programming Interface*(API) yang disediakan oleh organisasi Masyarakat Anti Fitnah Indonesia (Mafindo) melalui platform *TurnBackHoax.ID*. Proses pengumpulan data dilakukan menggunakan modul *requests* pada Python, mencakup rentang waktu dari tahun 2017 hingga 27 September 2024. Dari proses tersebut, terkumpul sebanyak 22.712 entri data yang kemudian melalui tahapan pembersihan sebagai berikut:

1. Menghapus nilai *null* dan data duplikat
2. Menghilangkan format tertentu pada kolom judul, seperti:
    - `[SALAH]`
    - `[PENIPUAN]`
    - `[KLARIFIKASI]`
3. Menghapus kategori tidak relevan atau ambigu pada kolom *status*, seperti:
    - ”sebagian benar”
    - ”belum ada bukti”
    - ”dalam proses”
4. Mengurangi jumlah kolom dengan hanya menggunakan kolom `judul` dan `konten` sebagai atribut utama.
5. Memberikan label pada data:
    - Label `1` untuk berita hoaks.
    - Label `0` untuk berita fakta.

Setelah proses pembersihan selesai, jumlah data yang digunakan adalah 14.057 entri berita hoaks dan 495 entri berita fakta. Namun, analisis terhadap dataset menunjukkan adanya ketidakseimbangan yang signifikan, di mana berita fakta hanya mencakup sekitar 3.25% dari total data.

Untuk mengatasi masalah ketidakseimbangan ini, digunakan dataset *Indonesia News Dataset (2024)* yang tersedia di platform Kaggle [50] sebagai data tambahan untuk berita fakta. Dataset tersebut memuat artikel berita dari media nasional terpercaya, seperti Kompas, Tempo, dan Detik, dengan periode waktu antara Januari 2024 hingga September 2024. Sebanyak 45.299 entri berita fakta diambil dari dataset ini untuk menambahkan data berita fakta. Tahapan pembersihan pada dataset fakta meliputi:

1. Menghapus nilai *null* dan data duplikat
2. Menghapus format tertentu pada awal konteks berita, seperti:
   - ”KOMPAS.com”
   - ”TEMPO.CO”
   - ”detik.com”
3. Mengurangi jumlah kolom dengan hanya menggunakan kolom `judul` dan `konten`.
4. Memberikan label `0` untuk semua entri berita fakta.

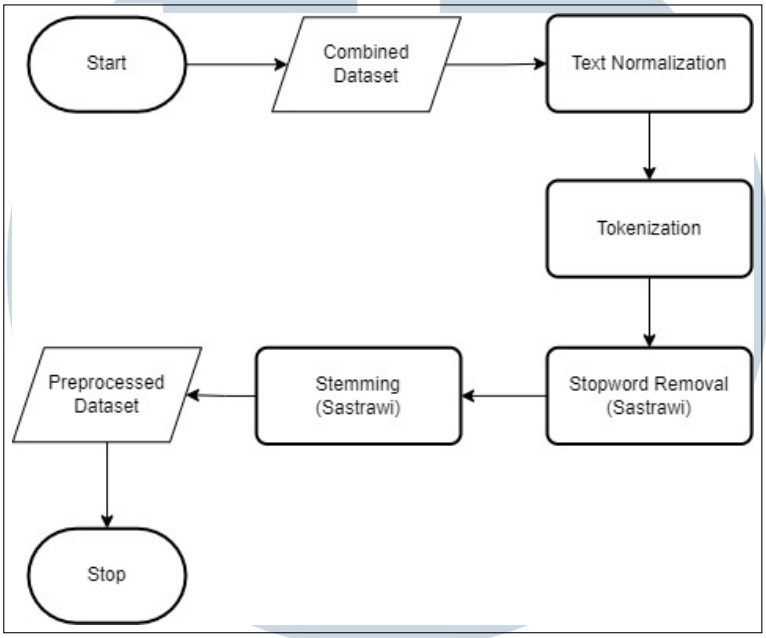
Dataset akhir yang digunakan dalam penelitian ini terdiri dari 59.847 entri, dengan rincian 14.056 data berita hoaks dan 45.299 entri berita fakta. Dataset yang telah digabungkan ini selanjutnya akan melalui *preprocessing* dan *feature extraction* menggunakan TF-IDF sebelum masuk ke pembuatan model LightGBM.

Tabel 3.2. Sumber dan Karakteristik Pengumpulan Data

Sumber Data	**Entri Hoaks**	**Entri Berita Asli**	**Total Entri**
API Mafindo	14.056	495	14.551
Dataset Kaggle	0	45.299	45.299
Total	**14.056**	**45.794**	**59.850**

3.4 Preprocessing Data

Data yang telah dikumpulkan selanjutnya diproses melalui beberapa tahapan sistematis. Tahapan tersebut terlampir pada Gambar 3.3.



Gambar 3.3. *Preprocessing*

1. Normalisasi Teks: Proses ini mencakup serangkaian langkah untuk membersihkan dan menstandarkan teks pada dataset, yang terdiri dari:
   (a) *Case Folding*: Mengubah seluruh teks menjadi huruf kecil untuk menstandarisasi format teks.
   (b) *Cleaning*: Menghapus elemen-elemen yang tidak relevan seperti:
      i. Emoji
      ii. URL dan tautan web
      iii. Tag HTML
      iv. Karakter spesial
      v. Angka
      vi. Spasi berlebih
2. Tokenisasi: Proses ini memecah teks yang telah dinormalisasi menjadi unit-unit lebih kecil berupa kata atau token. Sebagai contoh, teks berikut:

”Pemerintah sedang mempersiapkan rencana pembangunan infrastruktur besar-besaran.”

Setelah melalui proses tokenisasi, teks akan menjadi:

[”pemerintah”, ”sedang”, ”mempersiapkan”, ”rencana”, ”pembangunan”, ”infrastruktur”, ”besar-besaran”]

Token-token ini menjadi dasar untuk proses pengolahan teks selanjutnya.

3. Penghapusan Stopword: Menghapus kata-kata umum yang tidak memiliki kontribusi signifikan terhadap makna teks, seperti kata penghubung, kata depan, dan kata ganti. Penghapusan dilakukan menggunakan fungsi `StopWordRemover` pada modul Sastrawi. Daftar *Stop Words* yang digunakan terlampir pada Tabel 3.3.

Tabel 3.3. Daftar *Stop Words* yang Digunakan

yang	untuk	pada	ke	para	namun
menurut	antara	dia	dua	ia	seperti
jika	sehingga	kembali	dan	tidak	ini
karena	kepada	oleh	saat	harus	sementara
setelah	belum	kami	sekitar	bagi	serta
di	dari	telah	sebagai	masih	hal
ketika	adalah	itu	dalam	bisa	bahwa
atau	hanya	kita	dengan	akan	juga
ada	mereka	sudah	saya	terhadap	secara
agar	lain	anda	begitu	mengapa	kenapa
yaitu	yakni	daripada	itulah	lagi	maka
tentang	demi	dimana	kemana	pula	sambil
sebelum	sesudah	supaya	guna	kah	pun
sampai	sedangkan	selagi	sementara	tetapi	apakah
kecuali	sebab	selain	seolah	seraya	seterusnya
tanpa	agak	boleh	dapat	dsb	dst
dll	dahulu	dulunya	anu	demikian	tapi
ingin	juga	nggak	mari	nanti	melainkan
oh	ok	seharusnya	sebetulnya	setiap	setidaknya
sesuatu	pasti	saja	toh	ya	walau
tolong	tentu	amat	apalagi	bagaimanapun	

Sebagai contoh, teks setelah tokenisasi:

[”pemerintah”, ”sedang”, ”mempersiapkan”, ”rencana”, ”pembangunan”, ”infrastruktur”, ”besar-besaran”]

Setelah proses ini, hasilnya menjadi:

[”pemerintah”, ”mempersiapkan”, ”rencana”, ”pembangunan”, ”infrastruktur”]

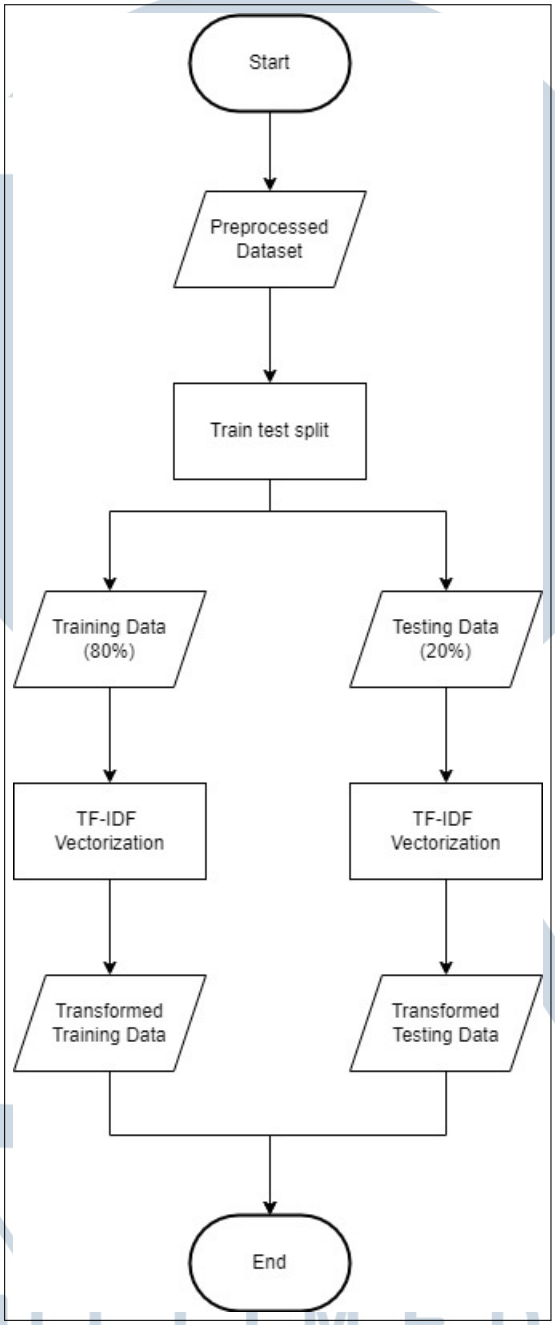
4. Stemming: Proses ini mengubah kata-kata menjadi bentuk dasarnya dengan menghilangkan imbuhan (awalan, sisipan, dan akhiran). Proses ini dilakukan menggunakan *library* Sastrawi Stemmer. Sebagai contoh:

   [”mempersiapkan”, ”pembangunan”, ”infrastruktur”]

   Setelah stemming, hasilnya menjadi:

   [”siap”, ”bangun”, ”infrastruktur”]

3.5 Pembagian Dataset dan Ekstraksi Fitur



Gambar 3.4. *Dataset Split & TF-IDF Vectorization*

Setelah melalui tahap preprocessing, dataset dibagi menjadi dua bagian, *training set* (80%), dan *testing set* (20%). Pembagian ini bertujuan untuk memastikan bahwa model dilatih dengan cukup data, dan dievaluasi secara objektif pada data yang belum pernah dilihat sebelumnya. Selanjutnya, proses

ekstraksi fitur dilakukan menggunakan metode Term Frequency-Inverse Document Frequency (TF-IDF). Metode ini merepresentasikan teks sebagai matriks dokumen-kata dengan bobot numerik yang mencerminkan kepentingan kata tertentu dalam dokumen tertentu relatif terhadap seluruh korpus.

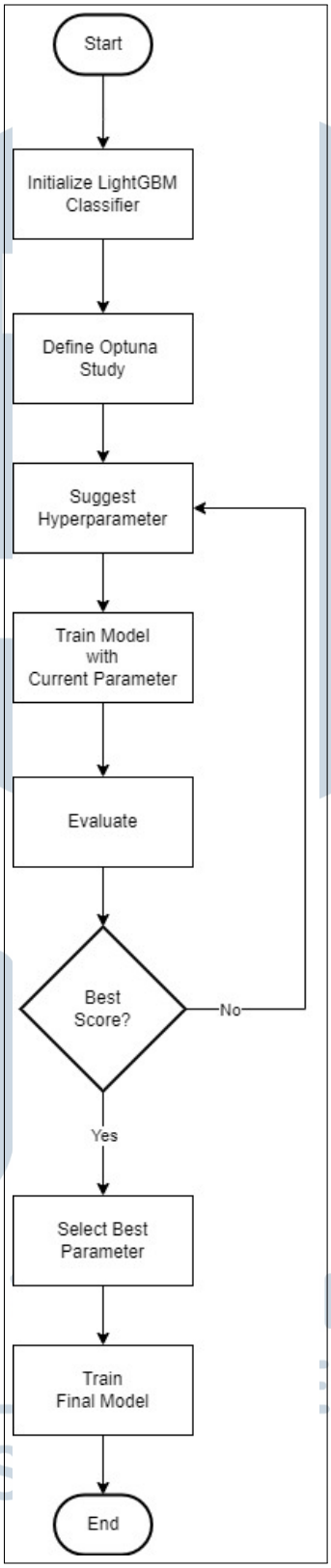
Tahapan penghitungan TF-IDF meliputi:

1. Term Frequency (TF): Menghitung frekuensi relatif kata dalam sebuah dokumen.
2. Inverse Document Frequency (IDF): Menghitung pentingnya kata berdasarkan jumlah dokumen di corpus yang mengandung kata tersebut.
3. TF-IDF Weighting: Mengalikan nilai TF dengan IDF untuk menentukan bobot akhir.

Proses ini menghasilkan matriks sparsa (sparse matrix) yang digunakan sebagai masukan (input) ke algoritma pembelajaran mesin. Matriks ini memungkinkan model untuk memanfaatkan informasi penting dari teks tanpa memasukkan redundansi atau informasi yang kurang relevan.

3.6 Perancangan Model LightGBM

Proses pembuatan model LightGBM dilakukan secara sistematis melalui beberapa tahapan, seperti yang ditunjukkan pada Gambar 3.5. Tahapan ini meliputi inisialisasi model dasar, optimasi parameter menggunakan framework Optuna, dan pelatihan model menggunakan parameter optimal.



Gambar 3.5. Tahapan Perancangan Model LightGBM

3.6.1 Inisialisasi Model

Tahapan awal dimulai dengan inisialisasi model LightGBM menggunakan konfigurasi dasar. Model dikonfigurasi sebagai classifier biner dengan parameter awal yang dirancang untuk menangani karakteristik dataset.

3.6.2 Optimasi Parameter

Proses optimasi parameter menggunakan Optuna untuk mencari konfigurasi optimal LightGBM. Optuna mengeksplorasi ruang parameter dengan mendefinisikan range pencarian yang luas berdasarkan tabel 3.4. Fungsi objektif yang digunakan adalah akurasi klasifikasi pada data validasi.

Tabel 3.4. Ruang Parameter untuk Optimasi LightGBM

Parameter	**Range**	**Deskripsi**
boosting_type	{gbdt, dart}	Tipe boosting yang digunakan, seperti Gradient Boosted Decision Tree (gbdt) atau Dropouts (dart).
data_sampling_strategy	{bagging, goss}	Strategi sampling data yang digunakan, misalnya bagging atau Gradient-based One-Side Sampling.
learning_rate	0.01 – 0.3	Kecepatan pembelajaran untuk setiap iterasi.
n_estimators	50 – 1000	Jumlah pohon yang akan dibangun.
max_depth	3 – 15	Kedalaman maksimum dari pohon keputusan.
num_leaves	31 – 255	Jumlah maksimum leaf nodes dalam pohon.
min_data_in_leaf	10 – 100	Jumlah data minimum yang diperlukan untuk membuat leaf.
feature_fraction	0.5 – 1.0	Proporsi fitur yang digunakan pada setiap iterasi boosting.
bagging_fraction	0.5 – 1.0	Proporsi data yang digunakan untuk setiap iterasi bagging.
lambda_l1	0.0 – 10.0	Parameter regulasi L1 untuk mencegah overfitting.
lambda_l2	0.0 – 10.0	Parameter regulasi L2 untuk mencegah overfitting.

Setelah proses optimasi parameter selesai, hasil konfigurasi optimal dari Optuna tidak digunakan secara langsung. Sebagai langkah lanjutan,

dilakukan serangkaian percobaan tambahan untuk memverifikasi dan mengevaluasi pentingnya masing-masing parameter. Tahap ini bertujuan untuk memastikan bahwa konfigurasi parameter yang dihasilkan tidak hanya optimal pada data validasi, tetapi juga memberikan performa yang konsisten pada data uji. Percobaan ini mencakup penyesuaian manual pada beberapa parameter utama, seperti `data_sampling_strategy`, untuk mengidentifikasi parameter yang memiliki pengaruh signifikan terhadap performa model. Selanjutnya, hanya parameter yang dianggap memiliki pengaruh penting yang digunakan untuk pelatihan akhir model.

3.6.3 Pelatihan Model

Setelah mendapatkan parameter optimal dari proses optimasi, model dilatih menggunakan konfigurasi tersebut. LightGBM menggunakan teknik GOSS (Gradient-based One-Side Sampling), sebuah strategi sampling eksklusif yang dirancang untuk meningkatkan efisiensi training tanpa mengorbankan akurasi model. Teknik ini mengoptimalkan proses pembelajaran dengan mempertimbangkan distribusi gradien pada data training.

Parameter yang dioptimalkan mencakup learning rate untuk mengontrol kecepatan pembelajaran, jumlah estimators untuk menentukan kompleksitas ensemble, serta berbagai parameter tree seperti `num_leaves` dan `max_bin` untuk mengoptimalkan struktur pohon keputusan dan penggunaan memori. Konfigurasi parameter ini secara keseluruhan dirancang untuk mencapai keseimbangan antara kemampuan model dalam menangkap pola data dan efisiensi komputasi.

3.7 Evaluasi Model

Tahap evaluasi bertujuan untuk mengukur kinerja model LightGBM dalam mendeteksi berita hoaks. Beberapa metrik evaluasi yang digunakan untuk menilai performa model adalah sebagai berikut:

1. Confusion Matrix: merupakan sebuah visualisasi yang menunjukkan distribusi prediksi model terhadap dua kelas (hoaks dan fakta). Matriks ini menggambarkan empat kategori utama: True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN). Confusion matrix memberikan visualisasi yang lebih rinci mengenai kesalahan yang dilakukan oleh model dalam mengklasifikasikan berita. dimana:

(a) (TP) (True Positive) adalah jumlah prediksi berita hoaks yang benar,

(b) (TN) (True Negative) adalah jumlah prediksi berita fakta yang benar,

(c) (FP) (False Positive) adalah jumlah prediksi berita hoaks yang salah,

(d) (FN) (False Negative) adalah jumlah prediksi berita fakta yang salah.

2. Precision: mengukur ketepatan model dalam mengklasifikasikan berita hoaks. Precision dihitung dengan rumus:

$$\text{Precision} = \frac{TP}{TP+FP}$$

Precision menunjukkan seberapa banyak berita yang diprediksi hoaks benar-benar hoaks.

3. Recall: mengukur kemampuan model dalam mendeteksi semua berita hoaks yang ada dalam dataset. Recall dihitung dengan rumus:

$$\text{Recall} = \frac{TP}{TP+FN}$$

Recall menunjukkan seberapa banyak berita hoaks yang berhasil terdeteksi oleh model.

4. Akurasi: mengukur persentase prediksi yang benar dari keseluruhan jumlah prediksi yang dilakukan oleh model. Akurasi dihitung menggunakan rumus:

$$\text{Akurasi} = \frac{TP+TN}{TP+TN+FP+FN}$$

5. F1-Score: merupakan harmonisasi antara Precision dan Recall, yang digunakan untuk memberikan gambaran umum kinerja model. F1-Score dihitung dengan rumus:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

F1-Score memberikan nilai yang seimbang antara Precision dan Recall, sehingga cocok digunakan ketika terdapat ketidakseimbangan antara jumlah kelas hoaks dan fakta.

Untuk menilai performa model secara lebih robust dan menghindari overfitting, digunakan teknik evaluasi cross-validation. Cross-validation merupakan

metode evaluasi model yang membagi data menjadi beberapa bagian (fold) secara acak, kemudian melakukan pelatihan dan pengujian model secara berulang pada setiap kombinasi fold. Dalam penelitian ini, digunakan 10-fold cross-validation, di mana data dibagi menjadi 10 bagian yang sama besar. Proses evaluasi dengan 10-fold cross-validation meliputi tahapan berikut:

1. Membagi data menjadi 10 fold secara acak.
2. Melakukan pelatihan model LightGBM pada 9 fold dan evaluasi pada 1 fold yang tersisa secara bergantian hingga setiap fold pernah menjadi data validasi.
3. Menghitung rata-rata metrik evaluasi dari 10 fold sebagai performa akhir model.

Penggunaan cross-validation memungkinkan penilaian performa model yang lebih objektif dan mengurangi risiko overfitting, karena model diuji pada berbagai kombinasi data yang berbeda. Metrik-metrik evaluasi dihitung pada setiap fold dalam cross-validation, kemudian dirata-ratakan untuk mendapatkan performa keseluruhan model. Hasil evaluasi dengan cross-validation akan digunakan untuk menentukan apakah model telah mencapai performa yang diinginkan dan siap untuk diterapkan dalam deteksi berita hoaks secara efektif. Selain itu, hasil cross-validation juga dapat digunakan sebagai acuan untuk melakukan perbaikan atau penyesuaian lebih lanjut pada model jika diperlukan.

3.8 Gap Penelitian

Tabel 3.5. Perbandingan Penelitian Terdahulu

Sumber	Dataset	Algoritma	Metrik
[51]	229 hoax news 426 fact news	Random Forest, Multinomial Naive Bayes, SVM	F1-Score: 0.98 (RF) 0.43(NB) 0.71 (SVM)
[52]	155 fact news 95 hoax news	Naive Bayes	Accuracy: 78.6% Hoax: Precision: 67.1%, Recall 89.4% Valid: Precision: 91.6%,Recall 71.4%
[53]	228 fact news 372 hoax news	Multi-Layer Perceptron (MLP)	Hoax: Precision: 0.84% Recall: 0.73 Macro Average: F1-Score: 0.82
[12]	250 fact news 250 hoax news	XGBoost	Akurasi: 89% Precision: 90% Recall: 80%
[54]	433 fact news 683 hoax news	SVM, Random Forest, Nearest Centroid, SGD, Decision Tree, Bagging, AdaBoost, Gradient Boosting, MLP ANN, k-NN	Random Forest: Accuracy: 89% SVM, Gradient Boosting, AdaBoost, SGD, Decision Tree: Accuracy:80%
[55]	100 fact news 100 hoax news	SVM, SGD, Logistic Regression, Naive Bayes	Logistic Regression: F1-Score: 90.9%

Berdasarkan analisis penelitian-penelitian sebelumnya yang ditampilkan pada Tabel 3.5, terdapat beberapa kesenjangan yang menjadi fokus dalam penelitian ini. Pertama, dari segi ukuran dataset, penelitian-penelitian terdahulu umumnya menggunakan dataset dengan skala yang relatif kecil, berkisar antara 200-1000 entri data. Misalnya, penelitian [52] hanya menggunakan 250 entri data (155 berita fakta dan 95 berita hoaks), sementara [54] menggunakan sekitar 1100 entri data. Penelitian ini mengatasi keterbatasan tersebut dengan menggunakan dataset yang jauh lebih besar, yaitu 59.850 entri data (45.794 berita fakta dan 14.056 berita hoaks), yang memungkinkan model untuk belajar dari variasi pola dan karakteristik berita yang lebih beragam.

Kedua, dari segi metodologi, penelitian-penelitian sebelumnya telah mengeksplorasi berbagai algoritma machine learning seperti Random Forest, SVM, Naive Bayes, dan XGBoost. Namun, belum ada yang menggunakan algoritma LightGBM untuk klasifikasi berita hoaks dalam konteks bahasa Indonesia. LightGBM, sebagai algoritma gradient boosting yang lebih efisien, menawarkan pendekatan baru dalam menyelesaikan permasalahan klasifikasi berita hoaks. Keunggulan LightGBM dalam hal kecepatan komputasi dan kemampuannya dalam menangani data yang tidak seimbang menjadi nilai tambah yang belum dieksplorasi dalam penelitian-penelitian sebelumnya.