KLASIFIKASI KATEGORI BUKU BERDASARKAN JUDUL MENGGUNAKAN ALGORITMA NAÏVE BAYES DAN LSTM (STUDI KASUS: UPA PERPUSTAKAAN UNIVERSITAS LAMPUNG)

(Skripsi)

Oleh

VEZAN HIDAYATULLAH

NPM 2115061114

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS LAMPUNG

2026

KLASIFIKASI KATEGORI BUKU BERDASARKAN JUDUL MENGGUNAKAN ALGORITMA NAÏVE BAYES DAN LSTM (STUDI KASUS: UPA PERPUSTAKAAN UNIVERSITAS LAMPUNG)

Oleh

VEZAN HIDAYATULLAH

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai
Gelar Sarjana Teknik

Pada

Jurusan Teknik Elektro
Program Studi S1 Teknik Informatika
Fakultas Teknik Universitas Lampung

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
2026

ABSTRAK

KLASIFIKASI KATEGORI BUKU BERDASARKAN JUDUL MENGGUNAKAN ALGORITMA NAÏVE BAYES DAN LSTM (STUDI KASUS: UPA PERPUSTAKAAN UNIVERSITAS LAMPUNG)

Oleh

Vezan Hidayatullah

UPA Perpustakaan Universitas Lampung merupakan pusat sumber informasi vital yang mengelola lebih dari 29.874 (dengan 161.529 eksemplar) judul buku. Namun, pengelolaan data bibliografi di perpustakaan ini menghadapi kendala serius akibat riwayat migrasi sistem informasi yang berulang, mulai dari Dynic, SLiMS, eLib, hingga proses migrasi ke Inlislite saat ini. Proses migrasi tersebut mengakibatkan hilangnya sebagian atribut metadata penting, khususnya data kategori buku. Kondisi ini menyulitkan pustakawan dalam melakukan pemulihan data kategori yang hilang, serta menghambat proses penentuan klasifikasi yang tepat bagi koleksi buku baru yang terus bertambah. Oleh karena itu, diperlukan model klasifikasi otomatis yang handal untuk mengatasi permasalahan metadata tersebut. Penelitian ini bertujuan untuk membangun dan membandingkan kinerja dua pendekatan algoritma klasifikasi teks, yaitu metode *machine learning Naive Bayes* dan metode *deep learning Long Short-Term Memory* (LSTM). Hasil pengujian menunjukkan bahwa model *Bidirectional*-LSTM (1 Layer) dengan FastText Embedding memberikan kinerja terbaik, mencapai akurasi sebesar 79,30% dan *F1-Score* 0,73. Model ini unggul secara signifikan sebesar 10,30% dibandingkan model *Deep Feature Weighting Naïve Bayes* (DFWNB) dengan TF-IDF yang hanya mencapai akurasi 69,00%. Superioritas Bi-LSTM didorong oleh kemampuannya menangkap konteks urutan kata, serta penggunaan FastText yang efektif menangani morfologi Bahasa Indonesia. Berdasarkan hasil tersebut, model *Deep Learning* mampu memberikan performa klasifikasi yang lebih baik dibandingkan metode *Machine Learning* konvensional dalam studi kasus klasifikasi judul buku ini.

Kata kunci: Klasifikasi Teks, Migrasi Sistem Perpustakaan, *Deep Feature Weighting Naïve Bayes,* TF-IDF, Bi-LSTM*, FastText.*

ABSTRACT

BOOK CATEGORY CLASSIFICATION BASED ON TITLES USING NAÏVE BAYES AND LSTM ALGORITHMS (CASE STUDY: UPA PERPUSTAKAAN UNIVERSITAS LAMPUNG)

By

Vezan Hidayatullah

UPA Perpustakaan Universitas Lampung is a vital information resource center managing over 29,874 book titles (with 161,529 copies). However, bibliographic data management in this library faces serious challenges due to a history of repeated information system migrations, ranging from Dynic, SLiMS, and eLib to the current migration process to Inlislite. These migration processes resulted in the loss of some crucial metadata attributes, specifically book category data. This condition makes it difficult for librarians to recover the lost category data and hinders the accurate classification process for the continuously growing collection of new books. Therefore, a reliable automated classification model is required to address these metadata issues. This research aims to build and compare the performance of two text classification algorithmic approaches: the Naive Bayes machine learning method and the Long Short-Term Memory (LSTM) deep learning method. The test results indicate that the Bidirectional-LSTM (1 Layer) model with FastText Embedding yielded the best performance, achieving an accuracy of 79.30% and an F1-Score of 0.73. This model significantly outperformed the Deep Feature Weighting Naïve Bayes (DFWNB) model with TF-IDF, which only reached an accuracy of 69.00%, with a performance gap of 10.30%. The superiority of Bi-LSTM is driven by its ability to capture word sequence context and the use of FastText, which effectively handles Indonesian morphology. Based on these results, the Deep Learning model provides better classification performance compared to conventional Machine Learning methods in this book title classification case study.

Keywords: Text Classification, Library System Migration*, Deep Feature Weighting Naïve Bayes,* TF-IDF, Bi-LSTM*, FastText.*

Judul Skripsi	: KLASIFIKASI KATEGORI BUKU BERDASARKAN JUDUL MENGGUNAKAN ALGORITMA NAÏVE BAYES DAN LSTM (STUDI KASUS: UPA PERPUSTAKAAN UNIVERSITAS LAMPUNG)
Nama Mahasiswa	: VEZAN HIDAYATULLAH
Nomor Pokok Mahasiswa	: 2115061114
Program Studi	: Teknik Informatika
Jurusan	: Teknik Elektro
Fakultas	: Teknik

MENYETUJUI

1. Komisi Pembimbing

Pembimbing Utama	Pembimbing Pendamping
Puput Budi Wintoro, S.Kom., M.T.I.	Rio Arjestia Pradipta, S.Kom, M.T.I
NIP. 198410312019031004	NIP. 198603232019031013

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Informatika
Herlinawati, S.T., M.T.	Yessi Mulyani, S.T., M.T.
NIP 197103141999032001	NIP 197312262000122001

MENGESAHKAN

1. Tim Penguji

Ketua : **Puput Budi Wintoro, S.Kom., M.T.I.** .....

Sekretaris : **Rio Ariestia Pradipta, S.Kom, M.T.I** .....

Penguji : **Ir. Titin Yulianti, S.T., M. Eng.** .....

2. Dekan Fakultas Teknik

Dr. H. Ahmad Herison, S.T.,M.T

NIP. 19691030 200003 1 001

Tanggal Lulus Ujian Skripsi: 21 Januari 2026

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini, menyatakan bahwa skripsi saya dengan judul "Klasifikasi Kategori Buku Berdasarkan Judul Menggunakan Algoritma Naïve Bayes Dan LSTM (Studi Kasus: UPA Perpustakaan Universitas Lampung)" dibuat oleh saya sendiri. Semua hasil yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti skripsi ini merupakan salinan atau di buat oleh orang lain saya bersedia menerima sanksi sesuai dengan ketentuan hukum dan akademik yang berlaku.

Bandar Lampung, 21 Januari 2026

Pembuat Pernyataan,

Vezam Hidayatullah

NPM. 2115061114

RIWAYAT HIDUP

Penulis bernama Vezan Hidayatullah, lahir di Bukittinggi pada tanggal 17 Juli 2002. Penulis merupakan anak ketiga dari tiga bersaudara dari pasangan Bapak Etmi Hardi dan Ibu Leni Hendrawati. Penulis mengawali pendidikan di TK Dharma Wanita UNP. Selanjutnya, penulis melanjutkan pendidikan dasar di SD Pembangunan UNP pada tahun 2009 hingga 2015. Pendidikan tingkat menengah pertama ditempuh di SMP Negeri 7 Padang pada tahun 2015 hingga 2018, dan tingkat menengah atas di SMA Negeri 2 Padang dari tahun 2018 hingga 2021. Pada tahun 2021, penulis diterima sebagai mahasiswa baru di Program Studi Teknik Informatika Jurusan Teknik Elektro Universitas Lampung melalui jalur Seleksi Bersama Masuk Perguruan Tinggi Negeri (SBMPTN). Selama menjadi mahasiswa penulis aktif mengikuti beberapa kegiatan, antara lain sebagai berikut:

1. Menjadi anggota biasa Himpunan Mahasiswa Teknik Elektro Universitas Lampung, Divisi Pengabdian Masyarakat dan Divisi Sosial tahun 2021-2023.
2. Menjadi anggota Sponsorship dalam acara *Electrical Engineering in Action* tahun 2023.
3. Mengikuti program Studi Independen Bersertifikat Kampus Merdeka Batch 5 di PT Semesta Integrasi Digital (Karier.mu) program Studi Independen *Data Analyst* pada Agustus sampai Desember 2023.
4. Melaksanakan Kuliah Kerja Nyata selama 36 hari di Kampung Way Tuba Asri, Kecamatan Way Tuba, Kabupaten Way Kanan, Lampung.
5. Mengikuti Kerja Praktek di Kementerian Komunikasi dan Informatika (Kominfo), Jakarta Pusat posisi Quality Assurance (QA) pada bulan Juli hingga Agustus 2024.

MOTTO

"Sesungguhnya bersama kesulitan ada kemudahan."

(QS. Al-Insyirah: 6)

"Kurang cerdas dapat diperbaiki dengan belajar, kurang cakap dapat dihilangkan dengan pengalaman. Namun tidak jujur sulit diperbaiki."

(Alm. Mohammad Hatta)

"Jika kamu tidak bertarung, kamu tidak akan menang."

(Attack on Titan – Eren Yeager)

"Tidak apa-apa tersesat, selama kamu terus berjalan."

(Penulis)

PERSEMBAHAN

Segala puji syukur penulis panjatkan ke hadirat Allah SWT atas limpahan rahmat, taufik, dan hidayah-Nya, sehingga penelitian sekaligus penyusunan skripsi ini dapat diselesaikan dengan baik.

Kupersembahkan skripsi ini kepada:

Kepada kedua orang tua, Ayah dan Ibu tercinta, yang dengan doa dan kasihnya senantiasa menjadi penopang, serta dengan segala dukungan dan pengorbanannya mengiringi perjalanan penulis menyelesaikan skripsi ini.

Diri saya sendiri yang telah berusaha dengan sebaik mungkin dalam menyelesaikan studi hingga menyelesaikan skripsi.

Serta almamater saya, Program Studi Teknik Informatika, Jurusan Teknik Elektro, Fakultas Teknik, Universitas Lampung.

SANWACANA

Alhamdulillahirabbil'alamin, segala puji bagi Allah Subhanahu Wa Ta'ala, Rabb semesta alam, Yang Maha Pengasih lagi Maha Penyayang. Shalawat dan salam semoga senantiasa tercurah kepada Baginda Rasulullah Muhammad shallallahu 'alaihi wa sallam, beserta keluarga, sahabat, dan seluruh pengikutnya yang setia mengikuti petunjuk-Nya. Aamiin. Atas izin dan ridha-Nya, penulis dapat menyelesaikan penulisan skripsi yang berjudul "Klasifikasi Kategori Buku Berdasarkan Judul Menggunakan Algoritma Naïve Bayes Dan LSTM (Studi Kasus: UPA Perpustakaan Universitas Lampung)" sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik pada Fakultas Teknik Universitas Lampung. Penyusunan skripsi ini tidak terlepas dari bantuan, bimbingan, saran, dan dukungan dari berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Bapak Dr. H. Ahmad Herison, S.T.,M.T., selaku Dekan Fakultas Teknik Universitas Lampung.
2. Ibu Herlinawati, S.T., M.T., selaku Ketua Jurusan Teknik Elektro Universitas Lampung.
3. Ibu Yessi Mulyani, S.T., M.T, selaku Ketua Program Studi Teknik Informatika Fakultas Teknik Universitas Lampung.
4. Ibu Resty Annisa, S.T., M.Kom selaku Dosen Pembimbing Akademik yang telah membantu atas arahan yang telah diberikan selama penulis menempuh pendidikan di Fakultas Teknik Universitas Lampung
5. Bapak Puput Budi Wintoro, S.Kom., M.T.I., selaku pebimbing utama, dan Bapak Rio Ariestia P, S. Kom. M.T.I, selaku pembimbing pendamping, yang dengan tanpa lelah dan bosan telah memberikan bimbingan, semangat dan mencurahkan waktunya yang demikian banyak dalam menyelesaikan skripsi ini.

6. Ibu Ir. Titin Yulianti, S.T., M.Eng. yang telah bersedia menjadi penguji dalam sidang skripsi.
7. Bapak Bayzoni, S.T., M.T. selaku Kepala UPT Perpustakaan Universitas Lampung, yang telah memberikan izin penelitian serta arahan dan saran berharga sehingga penelitian ini dapat terlaksana dengan baik.
8. Kedua orang tua, kakak dan seluruh keluarga yang tidak hentinya mendo'akan serta memberikan dorongan semangat dan materi.
9. Seluruh dosen dan karyawan/ti Fakultas Teknik Universitas Lampung khususnya Program Studi Teknik Informatika, terima kasih atas ilmu yang bermanfaat bagi penulis dan bantuan administratif yang diberikan kepada penulis selama menempuh pendidikan di Fakultas Teknik Universitas Lampung.
10. Almamater tercinta, Fakultas Teknik Universitas Lampung
11. Sahabat-sahabatku, Shelina, Bintang, Tasya yang memberikan semangat dan dukungan kepada penulis.
12. Teman-temanku, Rafi, Ridho Ramadhan, Rama, Panca, Ganang, Abdul, Farhat, Muhkito, Sandi, Pandu, Adil, Hazel, Milano, Andre, Dimas, Ridho Fauzi, Faiz yang telah menemani dan mewarnai perjalanan penulis selama menempuh pendidikan di Universitas Lampung.
13. Rekan-rekan mahasiswa di Program Studi Teknik Informatika yang telah memberikan saran, dukungan, dan bantuannya.
14. Pihak-pihak yang tidak dapat disebutkan satu persatu yang telah memberikan dukungan kepada penulis dalam penyusunan skripsi ini

Semoga Allah Subhanahu Wa Ta'ala, memberikan balasan yang baik atas jasa dan kebaikan yang telah diberikan kepada penulis. Akhir kata, penulis menyadari masih terdapat kekurangan dalam penulisan skripsi ini, akan tetapi penulis berharap semoga skripsi ini dapat berguna dan bermanfaat bagi semua pihak khususnya bagi pengembangan ilmu hukum pada umumnya.

Bandar Lampung, 2026

Penulis,

Vezan Hidayatullah

NPM. 2115061114

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

I. PENDAHULUAN

1.1 Latar Belakang

Perpustakaan merupakan salah satu pusat sumber informasi dan pengetahuan yang vital di lingkungan akademik, termasuk di Universitas Lampung. Menurut Undang-Undang Nomor 43 Tahun 2007 tentang Perpustakaan, perpustakaan memiliki peran penting dalam mencerdaskan kehidupan bangsa melalui penyediaan akses terhadap informasi dan pengetahuan.

Sebagai lembaga pendidikan, Universitas Lampung memiliki perpustakaan dengan koleksi kurang lebih 29.874 judul buku (dengan 161.529 eksemplar) yang terus bertambah setiap tahunnya. Namun, sistem peminjaman di perpustakaan ini telah mengalami tiga kali migrasi (dari Dynic, SLiMS, ke eLib) dan saat ini sedang dalam proses migrasi ke Inlislite. Proses migrasi yang berulang ini menyebabkan hilangnya beberapa informasi bibliografi, termasuk data kategori buku, Akibatnya pustakawan kesulitan dalam menentukan kembali data bibliografi buku yang hilang tersebut terfokus pada kategorinya, selain itu pustakawan juga kesulitan dalam menetapkan kategori yang tepat bagi koleksi buku baru yang belum terklasifikasi.

Oleh karena itu, salah satu solusi untuk mengatasi permasalahan ini adalah dengan memanfaatkan teknologi kecerdasan buatan (*artificial intelligence*/*AI*) melalui metode klasifikasi teks. Penerapan teknologi ini sejalan dengan perkembangan era digitalisasi dalam bidang pendidikan, yang menuntut peningkatan efisiensi dalam pengelolaan informasi. Menurut UNESCO, integrasi teknologi dalam manajemen perpustakaan merupakan langkah strategis dalam meningkatkan aksesibilitas, efisiensi, dan kualitas layanan, sehingga memungkinkan sistem perpustakaan untuk beradaptasi dengan kebutuhan pengguna secara lebih optimal.

Dalam klasifikasi teks, terdapat beberapa model yang dapat digunakan, baik model tradisional maupun model *deep learning*. Beberapa model tradisional yang sering

digunakan adalah *Naive Bayes, Support Vector Machine*(SVM), *K-nearest Neighbor*(KNN)*,* dan *Decision Tree*. Berdasarkan hasil perbandingan salah satu penelitian, *Naïve Bayes* memiliki kecepatan komputasi yang lebih unggul dibandingkan SVM dan KNN, dengan waktu pemrosesan 3,42 detik, jauh lebih cepat dibandingkan SVM yang memerlukan 484,75 detik dan KNN 19,60 detik. Meskipun SVM memiliki presisi lebih tinggi (95,7%) dibandingkan dengan *Naïve Bayes* (92,1%), keterbatasannya terletak pada waktu komputasi yang sangat lama, sehingga kurang efisien untuk dataset yang besar[1]. Oleh karena itu *Naïve Bayes* menjadi pilihan yang tepat karena menunjukkan keseimbangan antara kecepatan dan akurasi, terutama untuk klasifikasi teks buku perpustakaan. *Naive Bayes* memiliki beberapa varian, di antaranya *Gaussian Naive Bayes*, *Multinomial Naive Bayes*, dan *Bernoulli Naive Bayes*. Dalam penelitian ini, algoritma *Multinomial Naive Bayes* (MNB) digunakan dan dikombinasikan dengan TF-IDF (*Term Frequency-Inverse Document Frequency*). Kombinasi ini terbukti efektif untuk klasifikasi teks berdasarkan frekuensi kata, sehingga cocok untuk penelitian ini[2]. Di sisi lain, beberapa model deep learning yang biasa digunakan untuk klasifikasi teks adalah *Long Short-Term Memory* (LSTM), CNN, BERT. RNN dapat menangani despedensi dalam teks panjang dan mampu menangkap konteks kata dengan baik, sedangkan CNN kurang efektif jika menangani teks panjang. BERT sendiri membutuhkan daya komputasi yang lebih besar dibandingkan 2 algoritma lainnya[3]. Oleh karena itu LSTM menjadi pilihan yang tepat karena menunjukkan keseimbangan antara akurasi dan kecepatan komputasi dibandingkan CNN dan BERT dalam klasifikasi teks.

Dalam penelitian ini, performa kedua algoritma, yaitu MNB dan LSTM, akan dibandingkan untuk menentukan model terbaik yang dapat diimplementasikan dalam sistem klasifikasi kategori buku di UPA Perpustakaan Universitas Lampung. Perbandingan ini dilakukan dengan mengevaluasi metrik-metrik kinerja seperti akurasi, presisi, recall, dan F1-score. Model terbaik yang dipilih akan menjadi solusi untuk meningkatkan efisiensi dan akurasi dalam pengelolaan koleksi buku, sehingga mendukung layanan perpustakaan yang lebih baik bagi pengguna.

Penelitian ini diharapkan dapat membantu proses klasifikasi kategori buku di UPA Perpustakaan Universitas Lampung. Fokus utama adalah memanfaatkan data judul buku yang telah dikumpulkan dari perpustakaan sebagai data pelatihan, sehingga model dapat mempelajari pola-pola penting dalam menentukan kategori. Dengan menerapkan kedua algoritma ini, diharapkan model yang dihasilkan mampu meminimalkan kendala yang selama ini dihadapi dalam pengelolaan koleksi buku.

Berdasarkan uraian di atas, penelitian ini bertujuan untuk mengembangkan dan membandingkan model klasifikasi kategori buku menggunakan algoritma *Naïve Bayes* dan LSTM. Diharapkan, model terbaik yang dihasilkan tidak hanya membantu mempercepat proses klasifikasi buku tetapi juga meningkatkan akurasi dan konsistensi dalam pengelolaan koleksi perpustakaan. Dengan model yang dikembangkan, aksesibilitas pengguna terhadap buku yang relevan dapat ditingkatkan, sehingga mendukung peran UPA Perpustakaan Universitas Lampung sebagai unit pelayanan yang handal.

1.2 Perumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, beberapa masalah utama yang dapat dirumuskan dalam penelitian ini adalah sebagai berikut::

1. Bagaimana performa algoritma *Naïve Bayes* dan LSTM dalam mengklasifikasikan kategori buku berdasarkan judul buku di UPA Perpustakaan Universitas Lampung?
2. Model manakah yang lebih optimal untuk diimplementasikan dalam sistem klasifikasi kategori buku di UPA Perpustakaan Universitas Lampung: *Naïve Bayes* atau LSTM?

1.3 Tujuan Penelitian

Tujuan penelitian ini adalah:

1. Mengukur perbandingan performa algoritma *Naïve Bayes* dan LSTM dalam mengklasifikasikan kategori buku berdasarkan judul buku di UPA Perpustakaan Universitas Lampung.

2. Membandingkan model yang lebih optimal untuk diimplementasikan dalam sistem klasifikasi kategori buku di UPA Perpustakaan Universitas Lampung.

1.4 Batasan Masalah

Dalam penelitian ini, pembatasan masalah meliputi hal-hal sebagai berikut:

1. Data buku yang digunakan hanya mencakup judul dan kategori buku yang diperoleh dari Perpustakaan Universitas Lampung.
2. Klasifikasi yang dilakukan hanya 10 kategori utama berdasarkan DDC.
3. Proses klasifikasi hanya berdasarkan judul buku, tanpa mempertimbangkan informasi tambahan seperti sinopsis, nama pengarang, atau tahun terbit.
4. Penelitian hanya sebatas pengujian kualitas model

1.5 Sistematika Penulisan Skripsi

Sistematika penulisan skripsi / tugas akhir ini terdiri dari 5 (lima) bab sebagai berikut:

BAB I : PENDAHULUAN

Bab ini memuat latar belakang masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, dan juga batasan penelitian.

BAB II : TINJAUAN PUSTAKA

Bab ini memuat mengenai teori-teori dasar yang digunakan dalam penelitian untuk klasifikasi teks menggunakan algoritma *Naive Bayes* dan LSTM, serta penelitian-penelitian terdahulu yang digunakan sebagai referensi penelitian.

BAB III : METODOLOGI PENELITIAN

Bab ini membahas mengenai waktu dan tempat pelaksanaan dari penelitian yang dilaksanakan, alat dan bahan yang berisi software, hardware, dan data yang akan digunakan, serta metode penelitian

yang digunakan dalam penelitian untuk membuat rekomendasi user terhadap buku yang akan di pinjam selanjutnya.

BAB IV : PEMBAHASAN

Bab ini memuat mengenai hasil dan pembahasan yang diperoleh dalam penelitian.

BAB V : KESIMPULAN DAN SARAN

Bab ini memuat mengenai kesimpulan yang diperoleh dari hasil penelitian yang telah dilakukan serta saran dari penulis yang dapat digunakan untuk kedepannya.

II. TINJAUAN PUSTAKA

2.1 *Natural Language Processing* (NLP)

Natural Language Processing (NLP) adalah salah satu cabang ilmu AI yang berfokus dalam pengolahan bahasa alami[4]. Metode pemrosesan bahasa alami dirancang untuk secara otomatis memproses dan menganalisis sejumlah besar data tekstual. NLP bertujuan untuk membuat mesin agar dapat memahami makna bahasa manusia kemudian dapat memberi respon yang sesuai. NLP memungkinkan mesin untuk memahami, menafsirkan, dan berinteraksi dengan manusia dengan cara yang lebih alami[4]. Dalam konteks pengolahan teks, NLP mencakup teknik-teknik seperti tokenisasi, penghapusan *stopwords*, *stemming*, *lemmatization*, dan analisis sintaksis. Teknik-teknik ini membantu mengubah data teks mentah menjadi format yang dapat diolah oleh algoritma machine learning.

Perkembangan NLP digunakan untuk tugas-tugas seperti evaluasi kesamaan semantik, pengambilan informasi, klasifikasi teks, pengenalan entitas bernama, terjemahan mesin, analisis sentimen, dan pembuatan basis pengetahuan otomatis, meningkatkan interaksi manusia-mesin melalui analisis otomatis dan sintesis bahasa alami. Dalam perpustakaan, NLP memainkan peran penting dalam pengelolaan koleksi buku. Dengan NLP, judul buku dapat diubah menjadi vektor numerik yang dapat diproses oleh algoritma untuk tugas klasifikasi. Dengan mengotomatiskan pemrosesan data bahasa alami, NLP mendukung pengambilan keputusan yang terinformasi dan meningkatkan pengalaman pengguna, yang pada akhirnya mengarah pada efisiensi operasional yang lebih besar dalam sistem perpustakaan yang digerakkan oleh AI[5].

2.2 *Text Classification*

Klasifikasi teks adalah proses penting dalam NLP yang bertujuan untuk mengelompokkan dokumen teks ke dalam kategori atau label tertentu berdasarkan konten yang terkandung di dalamnya[6]. Perkembangan bidang ini telah ditinjau secara sistematis dengan memetakan tren klasifikasi teks dari tahun 2015 hingga 2025, termasuk metodologi yang digunakan serta berbagai tantangan dalam menangani beragam domain data [7]. Proses ini melibatkan beberapa langkah penting, termasuk pengumpulan data, pra-pemrosesan teks, ekstraksi fitur, dan penerapan algoritma pembelajaran mesin untuk membangun model klasifikasi. Dalam klasifikasi teks, setiap dokumen dianalisis untuk menentukan kategori mana yang paling sesuai, yang dapat mencakup berbagai topik seperti berita, ulasan produk, atau dokumen akademis. Contoh penerapan klasifikasi teks termasuk klasifikasi email spam, analisis sentimen, dan pengelompokan dokumen berita.

Dalam proses klasifikasi teks, representasi data menjadi salah satu elemen penting. Data teks mentah perlu diubah menjadi representasi numerik yang dapat dipahami oleh algoritma machine learning. Salah satu teknik yang sering digunakan adalah TF-IDF, yang mengukur seberapa penting sebuah kata dalam sebuah dokumen relatif terhadap keseluruhan koleksi dokumen. Representasi ini membantu algoritma untuk mengenali pola yang relevan dari data teks dan mengabaikan informasi yang tidak signifikan.

Klasifikasi teks telah berkembang pesat dengan kemajuan teknologi dan metode pembelajaran mendalam. Berbagai algoritma digunakan dalam proses ini, termasuk *Naive Bayes*, SVM, dan jaringan saraf dalam (*deep neural networks*). Kinerja model klasifikasi sangat bergantung pada kualitas fitur yang diekstraksi dari data dan teknik pemilihan fitur yang digunakan untuk mengurangi dimensi data[8].

Klasifikasi teks juga memiliki tantangan tersendiri, seperti data yang tidak seimbang, di mana jumlah dokumen dalam satu kategori jauh lebih banyak dibandingkan kategori lainnya. Hal ini dapat memengaruhi performa model jika tidak ditangani dengan benar. Teknik seperti sampling data atau pemberian bobot khusus pada kategori yang lebih jarang digunakan untuk mengatasi masalah ini.

Selain itu, *preprocessing* yang tidak optimal dapat menyebabkan hilangnya informasi penting, sehingga memengaruhi akurasi model. Oleh karena itu, langkah-langkah seperti *stemming* dan penghapusan *stopwords* harus dilakukan dengan hati-hati agar tidak menghilangkan informasi yang relevan.

Dalam penelitian ini, klasifikasi teks digunakan untuk mengelompokkan judul buku ke dalam kategori yang telah ditentukan. Dengan memanfaatkan algoritma *Naïve Bayes* dan LSTM, diharapkan sistem klasifikasi yang dihasilkan dapat meningkatkan efisiensi pengelolaan koleksi buku di Perpustakaan Universitas Lampung. Selain itu, penerapan klasifikasi teks ini juga memberikan kontribusi pada pengembangan sistem berbasis NLP dalam pengelolaan perpustakaan akademik.

2.3 Kategori Buku

Kategori buku merupakan bentuk pengelompokan bahan pustaka berdasarkan subjek, tema, atau bidang ilmu tertentu untuk memudahkan proses pencarian, penyimpanan, dan pengelolaan koleksi di perpustakaan[9]. Setiap buku diklasifikasikan ke dalam kategori tertentu sesuai dengan topik utama yang dibahas dalam isi buku tersebut. Dengan adanya sistem kategori, pustakawan dan pengguna dapat dengan mudah menemukan buku yang relevan dengan kebutuhan informasi mereka tanpa harus menelusuri seluruh koleksi yang ada.

Sebagian besar perpustakaan di Indonesia, termasuk UPA Perpustakaan Universitas Lampung, menggunakan sistem klasifikasi *Dewey Decimal Classification (DDC)* sebagai standar pengelompokan koleksi. Sistem ini membagi seluruh pengetahuan manusia menjadi sepuluh kelas utama, yang masing-masing dapat dipecah lagi menjadi subkelas yang lebih spesifik[9]. Sepuluh kategori utama DDC tersebut ditunjukkan pada Tabel berikut:

Tabel 2. 1 Kategori Utama DDC

Kelas	**Kategori Utama**
000	Komputer, informasi, dan referensi umum
100	Filsafat dan psikologi

Tabel 2. 1 (lanjutan)

200	Agama
300	Ilmu sosial
400	Bahasa
500	Sains dan matematika
600	Teknologi
700	Kesenian dan rekreasi
800	Sastra
900	Sejarah dan geografi

Klasifikasi kategori buku memiliki peran penting dalam sistem informasi perpustakaan, baik secara fisik maupun digital. Tujuan utama dari adanya kategori buku adalah:

1. Mempermudah penelusuran informasi, karena pengguna dapat mencari buku berdasarkan bidang ilmu tertentu tanpa perlu mengetahui judul atau pengarang secara spesifik.
2. Meningkatkan efisiensi pengelolaan koleksi, karena pustakawan dapat dengan mudah mengatur, menata, dan memperbarui koleksi sesuai bidangnya.
3. Mendukung integrasi sistem informasi perpustakaan, terutama dalam proses digitalisasi dan otomasi, sehingga sistem dapat memberikan hasil pencarian yang akurat berdasarkan subjek.

Sebaliknya, jika data kategori buku tidak ada, maka akan muncul sejumlah permasalahan, antara lain:

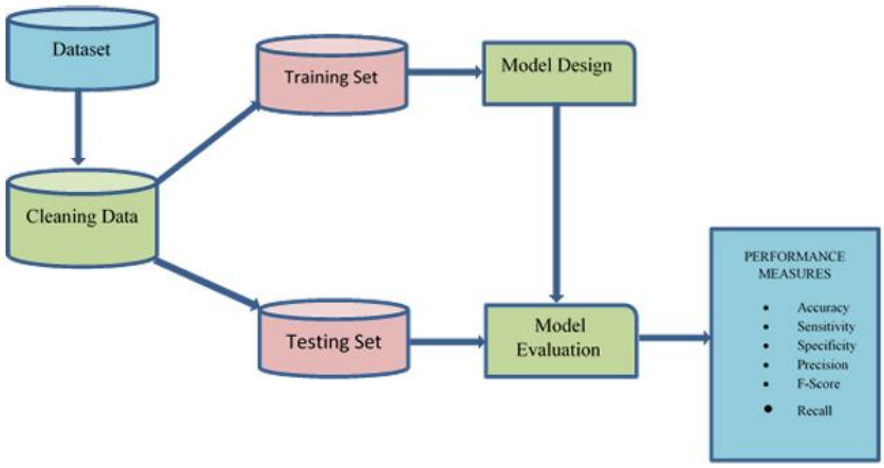
1. Ketidakefisienan dalam pengelolaan koleksi, misalnya penempatan buku yang tidak sesuai dengan bidangnya sehingga menghambat layanan pustaka.
2. Kesulitan dalam pencarian dan penelusuran buku, karena sistem tidak mampu memfilter koleksi berdasarkan bidang tertentu.

3. Menurunnya kualitas layanan pengguna, karena mahasiswa, dosen, maupun peneliti kesulitan menemukan referensi yang dibutuhkan meskipun sebenarnya koleksi tersebut tersedia.

Dalam konteks penelitian ini, kategori buku digunakan sebagai label target dalam proses klasifikasi teks berdasarkan judul buku. Dengan membangun sistem klasifikasi otomatis menggunakan metode *machine learning*, pustakawan dapat terbantu dalam memperbarui atau melengkapi data bibliografi buku yang belum memiliki kategori. Selain itu, sistem ini juga berpotensi mempercepat proses pengelolaan koleksi dan meningkatkan efisiensi layanan informasi di perpustakaan Universitas Lampung.

2.4 *Naïve Bayes*

Algoritma *Naïve Bayes* adalah salah satu metode data mining *supervised* berbasis probabilitas yang banyak digunakan untuk tugas klasifikasi berdasarkan pada penerapan teorema bayes[10]. Teorema bayes adalah perhitungan statistik dengan menghitung probabilitas kemiripan kasus lama yang ada di basis kasus dengan kasus baru. Algoritma ini disebut "naif" karena mengasumsikan bahwa semua fitur (atribut) bersifat independen satu sama lain, yang jarang terjadi dalam praktik[10]. Meskipun asumsi ini sederhana, *Naive Bayes* sering kali memberikan hasil yang baik dalam berbagai aplikasi, terutama dalam klasifikasi teks[10].



Gambar 2. 1 Alur Metode *Naive Bayes*[11]

Algoritma *Naïve Bayes* bekerja dengan membagi data menjadi *training set* dan *test set*. Model dikembangkan dengan melatih algoritma menggunakan *training set*, di mana probabilitas fitur dalam setiap kategori dihitung. Setelah model terbentuk, dilakukan evaluasi menggunakan test set untuk mengukur kinerjanya. Hasil evaluasi dianalisis menggunakan metrik untuk menentukan seberapa baik model dalam mengklasifikasikan data. *Naïve Bayes* bekerja dengan menghitung probabilitas suatu kategori berdasarkan fitur yang ada dalam data. Misalnya, untuk data teks, probabilitas suatu dokumen termasuk ke dalam kategori tertentu dihitung berdasarkan frekuensi kata-kata dalam dokumen tersebut. Proses ini didasarkan pada formula teorema Bayes[12]:

$$P(A|B) = \big(P(X|B) * P(A)\big) / P(B) \quad (1)$$

Keterangan:

- X: Data yang belum diketahui
- A: Hipotesis data
- P(A|B): Probabilitas suatu kategori A diberikan fitur B.
- P(B|A): Probabilitas fitur B muncul pada kategori A.
- P(A): Probabilitas awal kategori A (*prior*).
- P(B): Probabilitas keseluruhan fitur B (*evidence*).

Kelebihan dari algoritma *Naive Bayes* adalah kesederhanaan, kecepatan, dan efisiensinya dalam klasifikasi teks[13]. Algoritma ini sering menunjukkan akurasi tinggi dengan kumpulan data besar, menghindari bias subjektif dan *overfitting*, dan cocok untuk tugas klasifikasi multivariabel, membuatnya efektif untuk diagnosis kesalahan. Algoritma ini memerlukan sumber daya komputasi yang rendah[13], sehingga cocok untuk dataset besar seperti judul buku di perpustakaan. Selain itu, *Naive Bayes* dapat bekerja dengan baik bahkan ketika data yang digunakan tidak seimbang. Namun, algoritma ini memiliki beberapa kelemahan, termasuk asumsi independensi fitur, sensitivitas terhadap data yang hilang, dan ketidakmampuan untuk menangkap korelasi kompleks antar variabel. Keterbatasan ini dapat mempengaruhi kinerjanya dalam aplikasi dan skenario data tertentu[14].

Naïve Bayes memiliki beberapa varian, di antaranya:

2.4.1 *Multinomial Naïve Bayes*

Multinomial Naïve Bayes (MNB) adalah varian dari algoritma *Naïve Bayes* yang dirancang khusus untuk data diskrit, seperti frekuensi kata dalam dokumen teks[15]. Algoritma ini bekerja dengan menghitung probabilitas suatu dokumen termasuk dalam kategori tertentu berdasarkan frekuensi kemunculan kata dalam dokumen tersebut. MNB mengasumsikan bahwa setiap fitur (kata) bersifat independen, meskipun asumsi ini jarang terjadi dalam praktik, algoritma ini tetap menunjukkan performa yang baik dalam tugas klasifikasi teks.

Keunggulan utama MNB adalah kesederhanaan, kecepatan, dan efisiensinya dalam memproses data teks. Dalam penelitian ini, MNB dipilih karena kemampuannya untuk menangani data teks dengan efektif dan efisien, sehingga cocok untuk klasifikasi kategori buku berdasarkan judul. MNB akan menghitung probabilitas suatu judul buku termasuk dalam kategori tertentu berdasarkan frekuensi kata-kata yang muncul dalam judul tersebut

Selain algoritma MNB standar, terdapat beberapa pengembangan model yang bertujuan untuk mengatasi kelemahan asumsi independensi fitur dan meningkatkan akurasi klasifikasi, di antaranya adalah:

1. *Class-Specific Deep Feature Weighting* (CRaic)
   Pendekatan MNB standar sering kali mengasumsikan bobot fitur yang sama untuk semua kelas (*general feature weighting*). Ruan et al. (2020)[16] mengusulkan metode *Class-Specific Deep Feature Weighting* yang memberikan bobot spesifik untuk setiap fitur terhadap kelas tertentu. Metode ini menggunakan metrik statistik baru yang disebut CRaic. Rumus perhitungannya melibatkan faktor distribusi kelas dan frekuensi kelas dokumen:

$$CR_{a_i c} = R_{a_i c} \frac{p(a_i, c)}{p(a_i)} \ln (2 + \frac{n_c}{k_{a_i}}) \quad (2)$$

Keterangan:

- R_{a_ic} = Metrik statistik dasar relasi istilah-kelas.
- $p(a_i, c)$ = Probabilitas dokumen mengandung kata a_i dan berada di kelas c
- n_c = Jumlah total kelas.
- k_{a_i} = Jumlah kelas yang mengandung a_i .Semakin tinggi nilai CR_{a_ic}, semakin kuat ketergantungan positif antara kata tersebut dengan kelasnya.

2. *Log-likelihood Naïve Bayes* (LNB)

   LNB merupakan varian yang dikembangkan melalui pendekatan optimasi non-linear untuk menentukan bobot atribut terbaik. Kim & Lee (2022)[17] mengusulkan LNB dengan meminimalkan fungsi *loss* berbasis *log-likelihood* binomial. LNB dirancang agar lebih stabil terhadap data *noisy*. Rumus optimasi untuk mencari bobot pada LNB dinyatakan sebagai berikut:

$$w_{LNB} = \arg\min_{w} \sum_{i=1}^{n} \log[1 + \exp\{-y_i(P_0 + wP_{xi})\}] \qquad (3)$$

   Keterangan:

   - n = Jumlah data latih.
   - y_i = Label kelas (+1 atau -1).
   - P_0 = *Log-odds* dari probabilitas *prior*.
   - w = Vektor bobot yang dicari.
   - P_{xi} = Vektor *log-odds* dari probabilitas fitur.

3. *Generalized Deviance Naïve Bayes* (GDNB)

   GDNB merupakan pengembangan lebih lanjut dari LNB yang diperkenalkan oleh Kim & Lee (2022)[17]. Jika LNB menggunakan *log-likelihood* standar, GDNB memperkenalkan parameter penala (*tuning parameter*) yang disebut α (alpha) ke dalam fungsi *loss*. Fungsi objektif yang diminimalkan untuk mendapatkan bobot optimal pada GDNB adalah:

$$w_{GDNB} = \arg\min_{w} \sum_{i=1}^{n} \log[1 + \exp\{-\alpha y_i (P_0 + wP_{xi})\}] \quad (4)$$

Keterangan:

- α = Parameter penala (*tuning parameter*) untuk mengatur penalti terhadap *misclassification*.
- Nilai α dan bobot w dicari secara bersamaan menggunakan metode optimasi *non-linear* L-BFGS-B.

4. *Deep Feature Weighting Naïve Bayes* (DFWNB)

   Mayoritas metode pembobotan fitur tradisional hanya menerapkan bobot pada rumus akhir klasifikasi *Naïve Bayes* tanpa mengubah estimasi probabilitas bersyaratnya. Jiang et al. (2016)[18] memperkenalkan *Deep Feature Weighting* (DFW) yang mengintegrasikan bobot fitur tidak hanya pada aturan keputusan, tetapi juga masuk ke dalam perhitungan estimasi probabilitas bersyarat ($P(a_i|c)$) itu sendiri. Rumus estimasi probabilitas bersyarat pada DFWNB adalah:

$$P(a_i|c, W_i) = \frac{\sum_{j=1}^{n} W_i \delta(a_{ji}, a_i)\delta(c_j, c) + 1}{\sum_{j=1}^{n} W_i \delta(c_j, c) n_i} \quad (5)$$

   Keterangan:

   - W_i = Bobot fitur ke-i (bernilai 2 jika terpilih oleh *Correlation-based Feature Selection*/CFS, dan 1 jika tidak)
   - a_{ji} = Nilai fitur ke-i pada dokumen ke-j.
   - δ = Fungsi biner (bernilai 1 jika parameter cocok, 0 jika tidak).
   - n_i = Jumlah nilai unik dari fitur ke-i.

2.4.2 *Bernoulli Naïve Bayes*

Bernoulli Naïve Bayes adalah varian dari algoritma *Naïve Bayes* yang dirancang khusus untuk data biner (0 atau 1)[15]. Algoritma ini cocok untuk data teks yang direpresentasikan sebagai vektor biner, di mana setiap fitur (kata) menunjukkan ada (1) atau tidak ada (0) dalam dokumen. *Bernoulli Naïve Bayes* mengasumsikan bahwa setiap fitur mengikuti distribusi *Bernoulli*, sehingga lebih fokus pada keberadaan kata daripada frekuensinya. Meskipun kurang cocok untuk data teks

dengan frekuensi kata yang bervariasi, algoritma ini efektif untuk tugas klasifikasi biner, seperti deteksi spam atau analisis sentimen.

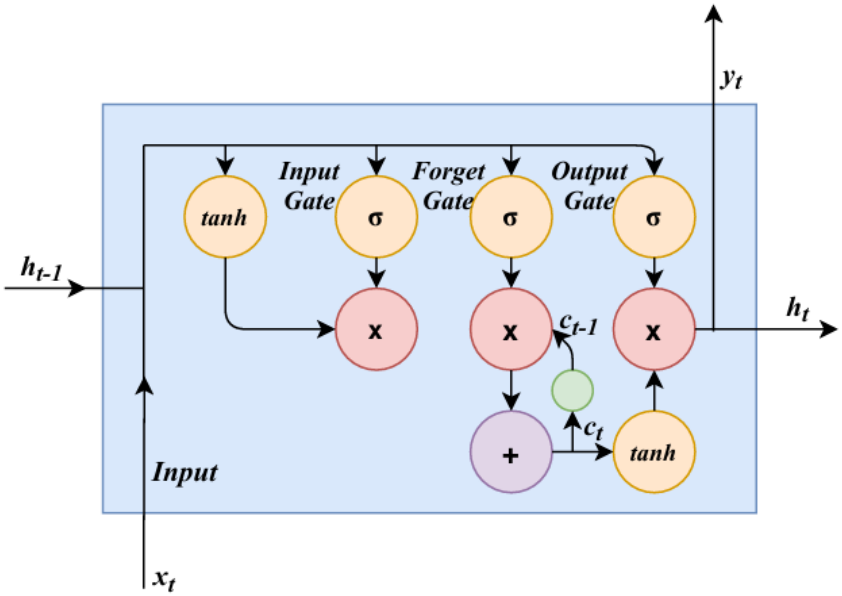
2.4.3 *Gaussian Naïve Bayes*

Gaussian Naïve Bayes adalah varian *Naive Bayes* yang digunakan untuk data kontinu atau numerik, di mana setiap fitur diasumsikan mengikuti distribusi *Gaussian* (normal)[15]. Algoritma ini menghitung probabilitas berdasarkan mean dan varians dari setiap fitur dalam setiap kategori. *Gaussian Naïve Bayes* cocok untuk data yang memiliki karakteristik numerik, seperti pengukuran atau skor, tetapi kurang efektif untuk data teks yang bersifat diskrit Meskipun demikian, algoritma ini sering digunakan dalam tugas klasifikasi yang melibatkan data numerik, seperti klasifikasi penyakit berdasarkan hasil tes medis.

2.5 *Long Short-Term Memory* (LSTM)

Long Short-Term Memory (LSTM) adalah arsitektur jaringan saraf tiruan rekursif (*Recurrent Neural Network* - RNN) yang dirancang khusus untuk mengatasi masalah gradien menghilang (*vanishing gradient*) yang sering terjadi pada RNN standar saat mempelajari urutan data yang panjang. Arsitektur ini pertama kali diperkenalkan oleh Hochreiter & Schmidhuber (1997) dengan tujuan untuk menangkap ketergantungan jangka panjang (*long-term dependencies*) dalam data sekuensial yang sulit dilakukan oleh RNN tradisional[19].

Berbeda dengan RNN standar yang hanya memiliki satu lapisan *tanh* sederhana dalam modul pengulangnya, LSTM memiliki struktur yang lebih kompleks yang disebut sebagai sel memori (*memory cell*). Kunci utama dari LSTM adalah status sel (*cell state*) yang berfungsi sebagai jalur informasi utama yang mengalir sepanjang rantai urutan dengan interaksi linear yang minimal, memungkinkan informasi untuk dipertahankan dalam jangka waktu yang lama tanpa mengalami distorsi signifikan. Mekanisme pengaturan aliran informasi di dalam sel LSTM dikendalikan oleh struktur yang disebut gerbang (*gates*). Setiap gerbang terdiri dari lapisan jaringan saraf *sigmoid* dan operasi perkalian *pointwise*. Menurut Goodfellow et al. (2016), arsitektur standar LSTM terdiri dari tiga gerbang utama, yaitu *Forget Gate*, *Input Gate*, dan *Output Gate*[20].



Gambar 2. 2 Arsitektur Sel LSTM[21].

Arsitektur LSTM bekerja dengan tiga gerbang utama. Setiap gerbang menggunakan lapisan *fully connected* (FC) dengan fungsi aktivasi *sigmoid* dan *tanh* untuk mengatur pembaruan nilai. LSTM mempertahankan informasi penting dalam urutan waktu dengan operasi perkalian dan penjumlahan elemen, memungkinkan model untuk menangani data sekuensial dengan lebih baik dibandingkan jaringan saraf konvensional. Ada beberapa langkah atau tahapan pada LSTM untuk memperoses inputan datanya yaitu:

1. *Forget Gate* (f_t)

   Gerbang ini menentukan informasi mana dari status sel sebelumnya (c_{t-1}) yang harus dibuang atau dilupakan. Keputusan ini dibuat oleh lapisan sigmoid yang melihat input saat ini (x_t) dan output sebelumnya (h_{t-1}). Konsep gate bertujuan untuk memungkinkan LSTM mereset status memorinya secara adaptif. Persamaannya adalah:

   $$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_i) \quad (6)$$

   Keterangan:

   - f_t = *Forget Gate*
   - σ = Fungsi Aktivasi *Sigmoid*

- W_f = *Weight Matrix*
- $[h_{t-1}, x_t]$ = Gabungan dari *Hidden State* sebelumnya dan Input saat ini.
- b_i = Bias Vector yang dipelajari

2. *Input Gate* (i_t)

   Pada tahap ini akan memilah dan menentukan informasi tertentu yang akan diperbarui ke bagian cell state dengan menggunakan fungsi aktivasi sigmoid.

$$i_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_i)$$
$$\tilde{c}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (7)$$

   Keterangan:

   - i_t = *Input Gate*
   - $\tilde{c}_t$ = *Candidate Cell State*
   - $tanh$ = Fungsi Aktivasi *Tanh*
   - W_c = *Weight Matrix* untuk *Cell State*
   - b_c = Bias Vector untuk *Cell State*

3. *Cell State* (c_t)

   Proses untuk memperbarui cell state lama c_{t-1} menjadi *cell state* baru dengan persamaan:

$$c_t = f_t . c_{t-1} + i_t . \tilde{c}_t \quad (8)$$

   Keterangan:

   - c_t = *Cell State* Baru

4. *Output Gate* (o_t)

   Proses menjalankan *sigmoid* untuk menghasilkan nilai *output* pada *hidden state* dan menempatkan *cell state* pada *tanh*, kemudian menghasilkan nilai *output sigmoid* dan nilai *output tanh* kedua hasil aktivasi tersebut dilakukan perkalian sebelum menuju langkah selanjutnya.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t . \tanh(c_t) \quad (9)$$

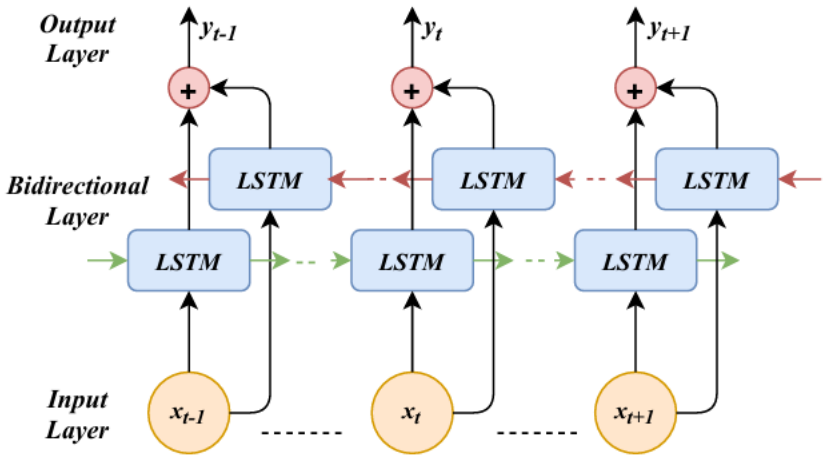
Keterangan:

- o_t = *Output Gate*
- W_o = *Weight Matrix* untuk *Output Gate*.
- b_o = Bias Vector untuk *Output Gate*.
- h_t = *Hidden State* Baru

Struktur ini memungkinkan LSTM untuk mempertahankan informasi penting dalam jangka panjang dan menghilangkan informasi yang tidak relevan, sehingga sangat efektif untuk tugas-tugas seperti klasifikasi teks. LSTM telah banyak digunakan dalam tugas klasifikasi teks karena kemampuannya untuk memahami konteks dan urutan kata dalam teks. Misalnya, dalam klasifikasi kategori berita, LSTM dapat mempelajari pola kata-kata yang muncul secara berurutan dan menghubungkannya dengan kategori tertentu[22].

Dalam penelitian ini, LSTM digunakan untuk mengklasifikasikan kategori buku berdasarkan judul buku. Dengan memanfaatkan kemampuan LSTM dalam memahami konteks dan urutan kata, diharapkan model dapat mempelajari pola-pola penting dalam judul buku dan mengklasifikasikannya ke dalam kategori yang sesuai. Meskipun LSTM memerlukan sumber daya komputasi yang besar, akurasi yang dihasilkan diharapkan dapat mengimbangi tantangan tersebut.

Lebih lanjut, arsitektur LSTM memiliki pengembangan yang dikenal sebagai Bidirectional LSTM (Bi-LSTM). Berbeda dengan LSTM standar yang hanya memproses urutan data dari satu arah (*forward*), Bi-LSTM memproses data dalam dua arah sekaligus, yaitu *forward* (dari awal ke akhir) dan *backward* (dari akhir ke awal). Dengan pendekatan ini, Bi-LSTM mampu memanfaatkan informasi dari konteks sebelumnya maupun sesudahnya secara bersamaan sehingga menghasilkan representasi fitur yang lebih kaya dan akurat [18].



Gambar 2. 3 Arsitektur Bi-LSTM[21].

Pada Bi-LSTM, untuk setiap input x_t, menghasilkan dua hidden state, yaitu *hidden state forward* $\overrightarrow{h_t}$dan *hidden state backward* $\overleftarrow{h_t}$. Keluaran akhir Bi-LSTM diperoleh melalui penggabungan kedua *hidden state* tersebut. Secara matematis, proses perhitungan pada Bi-LSTM dapat dijelaskan sebagai berikut[23]:

1. *Forward* LSTM:

$$\overrightarrow{h_t} = LSTMforward(x_t, \overrightarrow{h_t} - 1) \quad (10)$$

Keterangan:

- x_t = *Input* teks pada waktu ke-t.
- $\overrightarrow{h_t} - 1$ = *Hidden state* arah *forward* dari waktu sebelumnya.
- $\overrightarrow{h_t}$ = *Hidden state* arah *forward* pada waktu ke-t.

2. *Backward* LSTM:

$$\overleftarrow{h_t} = LSTMbackward(x_t, \overleftarrow{h_t} + 1) \quad (11)$$

Keterangan:

- $\overleftarrow{h_t} + 1$ = *Hidden state* arah *backward* dari waktu sebelumnya.
- $\overleftarrow{h_t}$ = *Hidden state* arah *backward* pada waktu ke-t.

3. Keluaran Bi-LSTM:

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \quad (12)$$

Keterangan:

- ; = Operasi penggabungan

Proses pada *forward* dan *backward* ini mencakup keseluruhan mekanisme gerbang pada LSTM, yaitu *forget gate*, *input gate*, *candidate cell*, dan *output gate*. Penggabungan ini membuat representasi fitur menjadi lebih kaya, karena menyatukan informasi yang datang dari kedua arah.

2.6 Komponen dan Konfigurasi LSTM

Dalam membangun model *deep learning* seperti LSTM, terdapat beberapa komponen matematis dan teknis yang krusial agar model dapat belajar dengan optimal, antara lain:

2.6.1 Fungsi Aktivasi (*Activation Function*)

Fungsi aktivasi adalah komponen krusial yang memperkenalkan non-linearitas ke dalam jaringan saraf, memungkinkan model untuk mempelajari fungsi yang kompleks dari data. Tanpa fungsi aktivasi, jaringan saraf, berapapun banyaknya lapisan (*layer*), hanya akan memiliki kapasitas representasi yang setara dengan model regresi linear sederhana[20].

Beberapa fungsi aktivasi yang umum digunakan antara lain[24]:

1. *Sigmoid Function*

   Fungsi *sigmoid* mengubah nilai *input* menjadi rentang antara 0 dan 1 dengan rumus:

   $$f(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{13}$$

   Keterangan:

   - $f(x)$ = Nilai Output *Sigmoid*
   - x = Input
   - e = Konstanta matematika

   Fungsi ini sering digunakan pada lapisan keluaran (*output layer*) untuk kasus klasifikasi biner karena menghasilkan nilai probabilitas tunggal. Namun, *sigmoid* cenderung mengalami masalah *vanishing gradient* saat digunakan pada jaringan yang dalam.

2. ReLU (*Rectified Linear Unit*)

   ReLU merupakan fungsi aktivasi yang sangat populer untuk lapisan tersembunyi (*hidden layer*), dengan rumus:

$$f(x) = max(0, x) \tag{14}$$

   Keterangan:

   - $f(x)$ = Nilai *output* ReLU
   - $max(0, x)$ = Mengambil nilai maksimum antara 0 dan x.

   Fungsi ini membuat pelatihan jaringan lebih cepat dan efisien serta membantu mengurangi masalah *vanishing gradient*. Dalam penelitian ini, fungsi ReLU digunakan pada lapisan *Dense* sebelum lapisan keluaran untuk menghasilkan representasi non-linear yang lebih kuat

3. *Softmax Function*

   Fungsi *softmax* digunakan pada lapisan keluaran untuk tugas klasifikasi multi-kelas. Fungsi ini mengubah hasil keluaran jaringan menjadi distribusi probabilitas, di mana jumlah total probabilitas seluruh kelas bernilai 1. Rumus fungsi *softmax* adalah:

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{15}$$

   Keterangan:

   - $f(x_i)$ = Probabilitas Kelas i
   - x_i = *Input* untuk Kelas i
   - e^{x_i} = Eksponensial dari *input*
   - $\sum_{j=1}^{n} e^{x_j}$ = Jumlah dari eksponensial semua input di semua kelas (j=1 sampai n). Fungsi ini berfungsi sebagai faktor normalisasi.

   Setiap nilai keluaran mewakili probabilitas suatu kelas. Dalam penelitian ini, fungsi *softmax* dipilih karena model LSTM digunakan untuk klasifikasi multi-kelas kategori buku, sehingga dibutuhkan fungsi aktivasi yang mampu memberikan perbandingan probabilitas antar kelas secara proporsional.

2.6.2 Teknik Regularisasi

Regularisasi adalah strategi untuk mengurangi kesalahan generalisasi (*test error*) tanpa meningkatkan kesalahan pelatihan (*training error*), dengan tujuan utama mencegah *overfitting*[20]. Untuk mengatasi hal ini, diterapkan teknik regularisasi:

1. *Dropout. Dropout* adalah metode regularisasi yang bekerja dengan cara menonaktifkan (membuang) unit neuron secara acak beserta koneksinya selama proses pelatihan dengan probabilitas tertentu p. Hal ini mencegah unit neuron untuk terlalu beradaptasi secara berlebihan terhadap data latih, sehingga memaksa jaringan untuk mempelajari fitur yang lebih kuat (*robust*)[25].
2. *Batch Normalization. Batch Normalization* adalah teknik untuk menormalkan input lapisan untuk setiap *mini-batch*. Metode ini bertujuan untuk mengatasi masalah *internal covariate shift*, yaitu perubahan distribusi input pada lapisan jaringan selama pelatihan. Dengan menstabilkan distribusi input, *Batch Normalization* memungkinkan penggunaan *learning rate* yang lebih tinggi dan mempercepat pelatihan[26]

2.6.3 Fungsi Loss (*Loss Function*)

Fungsi *loss* adalah metrik yang mengukur seberapa baik model memprediksi output yang benar[27]. Pemilihan *loss function* yang tepat akan membantu proses optimasi agar model belajar lebih efektif.

Untuk kasus klasifikasi multi-kelas, terdapat dua fungsi *loss* yang umum digunakan, yaitu[28]:

1. *Categorical Cross-Entropy* – digunakan ketika label target disajikan dalam bentuk *one-hot encoded vector*, di mana setiap kelas direpresentasikan sebagai vektor biner.
2. *Sparse Categorical Cross-Entropy* – digunakan ketika label target disajikan dalam bentuk integer (tanpa *one-hot encoding*).

Dalam penelitian ini, digunakan *Sparse Categorical Cross-Entropy* karena label kategori buku direpresentasikan sebagai nilai numerik (integer), bukan vektor biner.

Fungsi ini lebih efisien secara komputasi dan menghemat memori, terutama saat jumlah kelas cukup banyak. Fungsi *loss* ini bekerja dengan menghitung jarak antara probabilitas keluaran hasil fungsi *Softmax* dengan label integer sebenarnya, kemudian mengoptimalkan bobot jaringan agar kesalahan tersebut diminimalkan.

Kombinasi antara fungsi aktivasi *Softmax* dan loss function *Sparse Categorical Cross-Entropy* sangat umum digunakan dalam model *deep learning* untuk klasifikasi teks multi-kelas, karena keduanya saling melengkapi dalam proses probabilistik prediksi

2.6.4 Algoritma Optimisasi (*Optimizer*)

Optimizer adalah algoritma yang digunakan untuk memperbarui bobot model selama pelatihan[29]. Salah satu algoritma yang efektif untuk LSTM adalah RMSprop (*Root Mean Square Propagation*).

RMSprop bekerja dengan cara memodifikasi laju pembelajaran (*learning rate*) secara adaptif untuk setiap parameter. Algoritma ini membagi gradien dengan rata-rata bergerak dari kuadrat gradien terkini. Mekanisme ini mencegah laju pembelajaran menurun terlalu cepat yang sering terjadi pada algoritma, sehingga model dapat terus belajar dan mencapai konvergensi yang stabil, terutama pada data yang bersifat sekuensial atau *time-series.*

2.6.5 *Hyperparameter* Pelatihan

Dalam proses pelatihan model, terdapat parameter konfigurasi yang harus ditentukan secara manual sebelum pelatihan dimulai, yang dikenal sebagai *hyperparameter*. Dua parameter utama yang sangat memengaruhi dinamika pelatihan adalah[20]:

1. *Epoch*. *Epoch* didefinisikan sebagai satu putaran penuh di mana seluruh *dataset* pelatihan telah melewati jaringan saraf baik secara maju (*forward*) maupun mundur (*backward*). Jumlah *epoch* menentukan durasi pembelajaran model. Jumlah yang terlalu sedikit dapat menyebabkan *underfitting*, sedangkan jumlah yang terlalu banyak berpotensi menyebabkan *overfitting.*

2. *Batch Size*. *Batch size* adalah jumlah sampel data yang diproses oleh model dalam satu kali iterasi pembaruan bobot. Karena keterbatasan memori, data tidak diproses sekaligus melainkan dibagi menjadi *batch*. Ukuran *batch* yang lebih kecil memberikan estimasi gradien yang lebih bervariasi (stokastik) yang dapat membantu model menghindari *local minima*, namun membutuhkan waktu komputasi yang lebih lama per *epoch.*

2.7 *Data Collecting*

Data collecting atau pengumpulan data merupakan tahap penting dalam sebuah penelitian, khususnya pada studi berbasis *machine learning*. Data yang dikumpulkan harus mencerminkan karakteristik masalah yang akan diselesaikan serta relevan dengan tujuan penelitian[30]. Pengumpulan data yang efektif sangat penting karena kualitas model NLP sangat bergantung pada kualitas dan kuantitas data yang digunakan. Data yang baik dan representatif akan menghasilkan model yang lebih akurat dan dapat diandalkan dalam memproses dan menganalisis bahasa alami. Dalam konteks klasifikasi kategori buku, data berupa judul buku yang dikategorikan berdasarkan klasifikasi tertentu menjadi elemen utama untuk melatih dan menguji model algoritma. Data ini biasanya diperoleh dari sumber internal, seperti sistem manajemen perpustakaan, atau melalui proses ekstraksi manual dengan bantuan petugas perpustakaan.

2.8 *Exploratory Data Analysis* (EDA)

Exploratory Data Analysis (EDA) adalah pendekatan statistik untuk menganalisis kumpulan data guna merangkum karakteristik utamanya, sering kali dengan metode visual. Konsep ini dipopulerkan oleh John W. Tukey untuk mendorong pemahaman mendalam terhadap data sebelum melakukan pemodelan formal. Dalam konteks *data mining*, EDA berfungsi sebagai langkah awal untuk memastikan kualitas dan integritas data[31].

Berdasarkan karakteristik dataset perpustakaan yang digunakan, tahapan EDA dalam penelitian ini difokuskan pada dua aspek utama:

1. Analisis Kualitas Data (*Data Quality Analysis*)

   Kualitas data sangat menentukan kinerja model klasifikasi. Dua masalah utama yang sering ditemukan pada data teks mentah dan perlu diidentifikasi melalui EDA adalah:

   a. Redundansi (Duplikasi): Data duplikat dapat menyebabkan bias pada model karena model akan "menghafal" pola yang muncul berulang kali daripada mempelajarinya. Identifikasi proporsi data unik versus duplikat penting untuk menentukan seberapa agresif proses pembersihan data (*cleaning*) harus dilakukan[32].
   b. Inkonsistensi dan *Missing Values*: Data nyata sering kali tidak lengkap atau mengandung *noise* (nilai yang tidak valid). Analisis terhadap kelengkapan atribut (seperti kategori yang kosong atau format kode DDC yang tidak standar) diperlukan untuk memisahkan data yang layak diproses (*valid*) dan data yang harus dieliminasi[32].

2. Analisis Distribusi Kelas (*Class Distribution Analysis*)

   Analisis ini bertujuan untuk melihat sebaran jumlah data pada setiap kelas kategori target. Dalam klasifikasi teks, sering terjadi fenomena ketidakseimbangan kelas (*class imbalance*), di mana satu kategori memiliki jumlah sampel yang jauh lebih banyak (mayoritas) dibandingkan kategori lain (minoritas). Visualisasi distribusi kelas (misalnya menggunakan diagram batang) sangat krusial untuk mendeteksi potensi bias model. Jika ketimpangan data terlalu ekstrem, model cenderung akan memprediksi kelas mayoritas dan mengabaikan kelas minoritas, sehingga memerlukan strategi penanganan khusus seperti *undersampling* atau *oversampling* pada tahap persiapan data.

2.9 Text Preprocessing

Data teks mentah dari bahasa alami sering kali mengandung inkonsistensi, *noise*, dan struktur yang tidak beraturan yang dapat menurunkan kinerja algoritma pembelajaran mesin. Oleh karena itu, *Text Preprocessing* diperlukan untuk mentransformasi teks mentah menjadi format yang terstruktur dan bersih, serta mengurangi dimensi fitur tanpa menghilangkan informasi semantik yang penting

Berikut adalah teknik-teknik dalam preprocessing teks[30]:

1. *Tokenization*

   Teknik ini memecah teks menjadi unit-unit terkecil yang disebut token, seperti kata atau kalimat. *Tokenization* memungkinkan pemisahan elemen-elemen penting dalam teks sehingga dapat diolah secara terpisah.

2. *Lower Casing*

   Mengubah semua huruf dalam teks menjadi huruf kecil untuk menghindari perbedaan yang disebabkan oleh penggunaan huruf besar atau kecil. Hal ini memastikan konsistensi dalam pengolahan teks.

3. *Stopwords Removal*

   Stopwords adalah kata-kata umum yang memiliki frekuensi kemunculan tinggi namun minim makna semantik (kata fungsi), seperti kata hubung ("dan", "atau") dan kata depan ("di", "ke").. Penghapusan *stopwords* bertujuan untuk mengurangi dimensi ruang fitur dan memfokuskan algoritma pada kata-kata yang memiliki konten informasi penting (*content words*)[33]. Penelitian ini menggunakan daftar *stopwords* standar dari pustaka NLTK untuk teks berbahasa Inggris dan daftar *stopwords* bahasa Indonesia yang relevan.

4. *Stemming*

   Stemming adalah proses morfologi untuk memetakan variasi bentuk kata (kata berimbuhan) ke bentuk kata dasarnya (*root word*). Tujuannya adalah untuk menyatukan varian kata yang memiliki makna dasar yang sama (misalnya "memakan", "dimakan" menjadi "makan"), sehingga kepadatan fitur meningkat[33]. Mengingat dataset terdiri dari judul buku yang mungkin menggunakan Bahasa Indonesia dan Bahasa Inggris, penelitian ini menggunakan dua algoritma berbeda:

   - Porter Stemmer: Digunakan untuk teks berbahasa Inggris, diimplementasikan melalui modul PorterStemmer dari NLTK. Algoritma ini bekerja dengan menghapus sufiks umum bahasa Inggris secara bertahap[34].

- Sastrawi Stemmer: Digunakan untuk teks berbahasa Indonesia. Pustaka Sastrawi menerapkan algoritma Nazief & Adriani yang dikembangkan khusus untuk menangani kompleksitas morfologi bahasa Indonesia, termasuk penghapusan prefiks, sufiks, konfiks, dan *kiltik* secara aturan baku[35].

Preprocessing sangat penting karena dapat meningkatkan kualitas data teks dengan menghilangkan noise dan redundansi yang dapat mengganggu proses analisis. Dengan *preprocessing* yang baik, akurasi hasil analisis juga dapat meningkat, memungkinkan model NLP untuk bekerja lebih efektif dengan data yang bersih dan terstruktur.

2.10 Representasi Teks (*Text Representation*)

Representasi teks adalah tahapan krusial dalam pemrosesan NLP untuk mentransformasi data teks tidak terstruktur menjadi format vektor numerik yang dapat diproses oleh algoritma pembelajaran mesin. Komputer tidak dapat memahami teks secara langsung, sehingga setiap dokumen atau kata harus dipetakan ke dalam ruang vektor. Pemilihan metode representasi teks sangat memengaruhi kinerja model klasifikasi[33].

Dalam penelitian ini, digunakan dua pendekatan representasi teks yang berbeda sesuai dengan karakteristik model yang dibangun, yaitu TF-IDF untuk model *Naïve Bayes* dan *Word Embeddings* untuk model LSTM.

2.10.1 Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF adalah metode pembobotan fitur statistik yang mengukur seberapa penting suatu kata (*term*) dalam sebuah dokumen relatif terhadap sekumpulan dokumen (*corpus*). Metode ini menghasilkan vektor yang bersifat jarang (*sparse vector*), di mana dimensi vektor sama dengan jumlah kosakata unik dalam korpus[36].

Dalam penelitian ini, proses pembobotan TF-IDF diimplementasikan menggunakan modul *TfidfVectorizer* dari pustaka *Scikit-learn*. Modul ini secara otomatis mengonversi koleksi dokumen mentah menjadi matriks fitur TF-IDF dengan langkah-langkah perhitungan sebagai berikut:

a. *Term Frequency* (TF)

TF mengukur seberapa sering sebuah kata (term) muncul dalam suatu dokumen tertentu. Prinsip dasarnya adalah semakin sering suatu kata muncul dalam dokumen, semakin representatif kata tersebut terhadap topik atau isi dokumennya. Dalam bentuk yang paling dasar (seperti yang diterapkan pada pengaturan standar pustaka *Scikit-Learn*), TF dihitung berdasarkan frekuensi mentah (*raw count*) dari kemunculan kata[36]. Secara matematis, rumus TF dinyatakan sebagai berikut:

$$TF(t,d) = f_{t,d} \tag{16}$$

Keterangan:

- $TF(t,d)$: Nilai *Term Frequency* untuk kata t dalam dokumen d.
- $f_{t,d}$: Jumlah kemunculan (frekuensi) kata t dalam dokumen d.

b. *Inverse Document Frequency* (IDF)

IDF berfungsi untuk mengukur seberapa banyak informasi yang diberikan oleh sebuah kata. Kata-kata yang muncul di hampir semua dokumen (seperti "dan", "yang") dianggap kurang informatif dan diberikan bobot rendah, sedangkan kata-kata yang jarang muncul (eksklusif pada dokumen tertentu) diberikan bobot tinggi. Secara teoritis, IDF dihitung menggunakan logaritma dari rasio total dokumen (N) terhadap jumlah dokumen yang mengandung kata $t(df_t)$. Namun, dalam implementasi standar pada pustaka Scikit-learn, rumus ini dimodifikasi dengan penambahan konstanta (*smoothing*) untuk mencegah pembagian dengan nol (jika kata tidak muncul dalam data latih)[37]:

$$IDF(t) = \log\left(\frac{1+N}{1+df(t)}\right) + 1 \tag{17}$$

Keterangan:

- $IDF(t)$: Nilai *Inverse Document Frequency* untuk kata t.
- N: Jumlah total dokumen dalam korpus (dataset).
- $df(t)$: *Document Frequency*, yaitu jumlah dokumen yang mengandung kata t.

- 1: Konstanta *smoothing* untuk menghindari *error* matematika dan memastikan kata yang muncul di semua dokumen tidak memiliki bobot nol mutlak.

c. Pembobotan Akhir (TF-IDF)

Nilai akhir bobot TF-IDF adalah hasil perkalian antara skor TF dan skor IDF. Nilai yang tinggi menunjukkan bahwa kata tersebut memiliki frekuensi kemunculan yang tinggi dalam dokumen tertentu, namun jarang muncul di dokumen lain dalam korpus. Hal ini menjadikan kata tersebut sebagai fitur pembeda yang kuat untuk keperluan klasifikasi kategori[33]. Rumus perhitungan akhirnya adalah:

$$TF - IDF(t,d) = TF(t,d) \times IDF(t) \quad (18)$$

Keterangan:

- $TF - IDF(t,d)$: Bobot akhir kata t dalam dokumen d.

Setelah bobot dihitung, vektor hasil TF-IDF biasanya dinormalisasi menggunakan L2 Norm (*Euclidean Norm*) untuk memastikan panjang vektor setiap dokumen bernilai 1, sehingga perbandingan antar dokumen menjadi lebih adil. TF-IDF memberikan bobot yang lebih besar pada kata-kata yang sering muncul dalam dokumen tertentu tetapi jarang muncul di dokumen lain. Hal ini membantu model fokus pada kata-kata yang memiliki nilai informatif tinggi.

Meskipun TF-IDF memiliki banyak keunggulan, metode ini juga memiliki beberapa keterbatasan. TF-IDF tidak mempertimbangkan urutan kata dalam teks, sehingga kehilangan konteks semantik yang mungkin penting dalam beberapa kasus. Selain itu, pada dataset kecil, nilai IDF dapat menjadi kurang bermakna karena distribusi kata-kata yang terbatas. Oleh karena itu, *preprocessing* teks yang baik, seperti *stemming* dan penghapusan *stopwords*, sangat penting untuk mendukung efektivitas TF-IDF. Dalam penelitian ini, TF-IDF digunakan bersama langkah-langkah *preprocessing* untuk memastikan data judul buku terstruktur dengan baik sebelum dimasukkan ke dalam model klasifikasi.

2.10.2 Word Embeddings

Word Embedding adalah teknik representasi teks di mana kata-kata atau frasa dari kosakata dipetakan ke dalam vektor bilangan riil. Berbeda dengan representasi tradisional seperti TF-IDF yang menghasilkan vektor jarang (*sparse*), *word embedding* menghasilkan vektor padat (*dense vector*) dengan dimensi yang jauh lebih rendah. Tujuan utamanya adalah untuk menangkap makna semantik, hubungan sintaksis, dan konteks antar kata, sehingga kata-kata yang memiliki makna serupa akan memiliki representasi vektor yang berdekatan dalam ruang vektor[38].

Ada beberapa jenis *word embedding* yang sering digunakan, seperti berikut:

1. GloVe (*Global Vectors for Word Representation*)

   GloVe adalah metode *unsupervised learning* untuk mendapatkan representasi vektor kata yang diperkenalkan oleh Pennington et al. dari Universitas Stanford. GloVe bekerja berdasarkan prinsip matriks ko-okurensi global (*global co-occurrence matrix*), yang menghitung seberapa sering pasangan kata muncul bersamaan dalam seluruh korpus teks[39]. Ide dasar GloVe adalah memanfaatkan rasio probabilitas kemunculan kata untuk menentukan hubungan semantik. Misalnya, probabilitas kata "es" muncul bersama "dingin" akan jauh lebih tinggi dibandingkan bersama "panas".

   Fungsi objektif GloVe dirancang untuk meminimalkan selisih antara hasil perkalian titik (*dot product*) dua vektor kata dengan logaritma probabilitas kemunculan bersama mereka. Rumusnya dinyatakan sebagai berikut:

   $$J = \sum_{ij=1}^{V} f(X_{ij})\left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2 \qquad (19)$$

   Keterangan:

   - V: Ukuran kosakata (*vocabulary size*).
   - X_{ij}: Jumlah berapa kali kata j muncul dalam konteks kata i.
   - w_i: Vektor kata untuk kata i.
   - $\tilde{w}_j$: Vektor konteks untuk kata j.

- $b_i, \tilde{b}_j$: Bias untuk kata i dan kata j.
- $f(X_{ij})$: Fungsi pembobot (*weighting function*) untuk meredam efek kata-kata yang sangat sering muncul (seperti *stopwords*) agar tidak mendominasi pelatihan.

2. FastText

FastText adalah pengembangan metode *embedding* yang diperkenalkan oleh *Facebook AI Research* (FAIR). Berbeda dengan GloVe yang menganggap kata sebagai satuan terkecil (*atomic*), FastText memecah setiap kata menjadi bagian-bagian karakter yang lebih kecil yang disebut *n-grams* atau *subword information*[40].

Sebagai contoh, dengan *n-gram* = 5, kata "makan" akan direpresentasikan sebagai kumpulan vektor karakter: <ma, mak, aka, kan, an> serta kata utuhnya <makan>. Vektor akhir dari sebuah kata adalah penjumlahan dari seluruh vektor *n-grams* penyusunnya. Fungsi penilaian (*scoring function*) s untuk memprediksi konteks c dari kata w didefinisikan sebagai:

$$s(w, c) = \sum_{g \epsilon G_w} z_g^T v_c \tag{20}$$

Keterangan:

- $s(w, c)$: Skor kesesuaian antara kata w dan konteks c.
- G_w: Himpunan *n-grams* yang muncul dalam kata w.
- z_g: Representasi vektor untuk setiap *n-gram* g.
- v_c: Representasi vektor untuk kata konteks c.

Pendekatan ini memberikan keunggulan signifikan, terutama untuk bahasa dengan morfologi kaya (banyak imbuhan) seperti Bahasa Indonesia. FastText mampu membentuk representasi vektor untuk kata-kata yang jarang muncul atau bahkan kata yang mengandung kesalahan penulisan (*typo*) dengan cara menggabungkan vektor dari sub-kata yang dikenali.

2.11 Model Evaluation

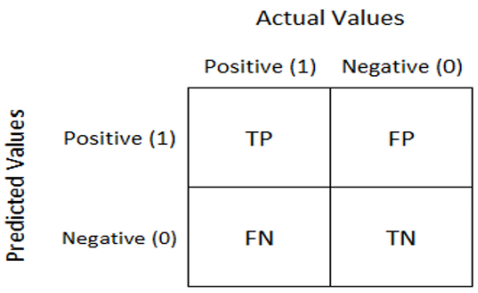
Evaluasi model adalah langkah penting dalam proses machine learning untuk menilai sejauh mana kinerja model tersebut setelah dibangun[41]. Dalam klasifikasi teks, evaluasi model bertujuan untuk mengukur seberapa baik model dapat

mengelompokkan data teks ke dalam kategori yang benar. Hasil evaluasi ini menjadi acuan untuk menilai apakah model sudah optimal atau perlu dilakukan perbaikan, seperti tuning parameter atau penyesuaian *preprocessing* data.

Dasar dari perhitungan metrik evaluasi klasifikasi adalah *Confusion Matrix*.

2.11.1 *Confusion Matrix*

Confusion Matrix adalah tabel representasi visual yang membandingkan hasil prediksi model dengan label kategori yang sebenarnya[42]. Matriks ini memberikan informasi detail mengenai jenis kesalahan yang dibuat oleh model, bukan hanya sekadar jumlah prediksi yang salah.



Gambar 2. 4 Confusion Matrix[42]

Dalam klasifikasi, terdapat empat kemungkinan hasil prediksi yang direpresentasikan dalam matriks ini:

1. *True Positive* (TP): Jumlah data kelas positif yang diprediksi secara benar sebagai kelas positif oleh model.
2. *True Negative* (TN): Jumlah data kelas negatif yang diprediksi secara benar sebagai kelas negatif oleh model.
3. *False Positive* (FP): Jumlah data kelas negatif yang salah diprediksi sebagai kelas positif (Kesalahan Tipe I).
4. *False Negative* (FN): Jumlah data kelas positif yang salah diprediksi sebagai kelas negatif (Kesalahan Tipe II).

Berdasarkan komponen-komponen dalam *Confusion Matrix* tersebut, kinerja model diukur menggunakan empat metrik utama, yaitu Akurasi, Presisi, *Recall*, dan *F1-Score*[42].

2.11.2 Akurasi (*Accuracy*)

Akurasi adalah proporsi total prediksi yang benar (baik positif maupun negatif) dibandingkan dengan total jumlah data. Secara matematis, akurasi dihitung dengan rumus:

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \quad (21)$$

2.11.3 Presisi (*Precision*)

Presisi mengukur tingkat ketepatan model dalam memprediksi kelas positif. Metrik ini menghitung perbandingan antara prediksi positif yang benar terhadap total seluruh hasil yang diprediksi sebagai positif. Presisi sangat penting dalam kasus di mana kesalahan *False Positive* memiliki dampak yang besar. Rumusnya adalah:

$$Precision = \frac{TP}{TP + FP} \times 100\% \quad (22)$$

2.11.4 *Recall* (Sensitivitas)

Recall, atau sering disebut juga sensitivitas, mengukur kemampuan model dalam menemukan kembali informasi atau mengenali seluruh data kelas positif yang ada. Metrik ini menghitung rasio prediksi positif yang benar terhadap total data yang sebenarnya positif. Rumusnya adalah:

$$Precision = \frac{TP}{TP + FN} \times 100\% \quad (23)$$

2.11.5 F1-Score

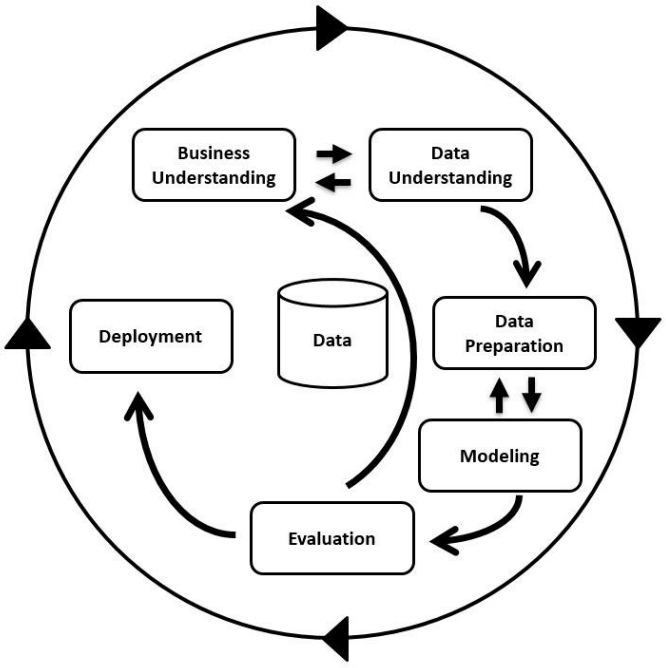
F1-Score adalah rata-rata harmonik dari presisi dan *recall*. Metrik ini memberikan gambaran kinerja yang lebih berimbang, terutama ketika terdapat ketimpangan distribusi kelas atau ketika penting untuk menyeimbangkan antara presisi dan

recall. Nilai *F1-Score* yang tinggi mengindikasikan bahwa model memiliki presisi dan *recall* yang baik secara bersamaan.

$$F1Score = \frac{Precision \times Recall}{Precision + Recall} \quad (24)$$

2.12 *Cross-Industry Standard Process for Data Mining* (CRISP-DM)

Cross-Industry Standard Process for Data Mining (CRISP-DM) adalah kerangka kerja standar yang menyediakan struktur metodologis untuk merencanakan dan melaksanakan proyek penggalian data (*data mining*). Metodologi ini dikembangkan oleh konsorsium industri pada tahun 1996 untuk mengubah *data mining* dari sekadar proses teknis menjadi proses bisnis yang terkelola dengan baik. CRISP-DM bersifat siklis (berulang) dan terdiri dari enam tahapan hierarkis yang saling terkait[43]. Sifat siklis dari CRISP-DM, khususnya hubungan timbal balik antara tahap pemahaman bisnis dan pemahaman data, serta antara pemodelan dan evaluasi, memungkinkan peneliti untuk terus menyempurnakan model hingga mencapai tujuan penelitian yang diharapkan.



Gambar 2. 5 Metodologi CRISP-DM[43]

2.12.1 *Business Understanding*

Tahap ini merupakan fase fundamental yang berfokus pada pemahaman tujuan proyek dari perspektif kebutuhan bisnis atau institusi. Tujuannya adalah mendefinisikan masalah spesifik yang ingin dipecahkan dan menerjemahkannya ke dalam definisi masalah teknis *data mining*[43]. Dalam penelitian ini, tahap ini mencakup identifikasi inefisiensi pada proses klasifikasi manual di perpustakaan dan penetapan tujuan untuk membangun sistem klasifikasi otomatis berbasis *Machine Learning*.

2.12.2 *Data Understanding*

Tahap ini dimulai dengan pengumpulan data awal, dilanjutkan dengan aktivitas untuk membiasakan diri dengan data, mengidentifikasi masalah kualitas data (seperti nilai yang hilang atau duplikasi), serta mendeteksi wawasan awal melalui statistik deskriptif atau visualisasi[43]. Pada penelitian ini, tahap ini melibatkan eksplorasi dataset judul buku, analisis distribusi kategori DDC, dan identifikasi karakteristik linguistik data.

2.12.3 *Data Preparation*

Tahap persiapan data sering kali menjadi fase yang paling memakan waktu. Fase ini mencakup semua kegiatan untuk mengonstruksi dataset final dari data mentah agar siap diproses oleh algoritma pemodelan[43]. Aktivitas utamanya meliputi pembersihan data (*data cleaning*), seleksi fitur, transformasi teks (*preprocessing*). Tahap ini secara spesifik meliputi langkah-langkah berikut:

1. **Pembersihan Data (*Cleaning Data*):** Proses mendeteksi dan mengoreksi data yang korup, tidak akurat, atau tidak lengkap. Ini mencakup penanganan nilai yang hilang (*missing values*), penghapusan duplikasi, dan perbaikan inkonsistensi data untuk menjamin kualitas input model.
2. **Pemilihan Fitur (*Feature Selection*):** Proses menyeleksi subset atribut yang paling relevan dan membuang atribut yang tidak memberikan kontribusi informasi signifikan. Tujuannya adalah untuk mengurangi dimensi data dan meningkatkan efisiensi komputasi model.

3. **Pra-pemrosesan (*Preprocessing*):** Tahap transformasi data ke dalam format yang dapat diterima oleh algoritma pemodelan. Pada data teks, ini melibatkan teknik seperti *lower case, tokenization*, dan *stemming*.

2.12.4 *Modeling*

Pada tahap ini, berbagai teknik pemodelan dipilih dan diterapkan, serta parameter-parameternya dikalibrasi (*hyperparameter tuning*) untuk mendapatkan nilai optimal. Karena beberapa teknik memiliki kebutuhan format data yang spesifik, sering kali terjadi perulangan (*looping*) kembali ke tahap persiapan data[43]. Dalam penelitian ini, fase ini mencakup pembangunan model menggunakan *Naïve Bayes* dan LSTM.

2.12.5 *Evaluation*

Sebelum model diterapkan, perlu dilakukan evaluasi mendalam untuk memastikan model tersebut tidak hanya memiliki akurasi statistik yang tinggi, tetapi juga memenuhi tujuan bisnis yang telah ditetapkan di awal. Dalam penelitian ini, fase ini mencakup perbandingan performa antara model Naïve Bayes dan LSTM untuk menentukan model terbaik.

2.12.6 *Deployment*

Pembentukan model bukanlah akhir dari proyek. Pengetahuan yang diperoleh dari data perlu diorganisir dan dipresentasikan sedemikian rupa sehingga dapat digunakan oleh pengguna akhir (*end-user*). Tahap ini dapat berkisar dari pembuatan laporan sederhana hingga implementasi proses *data mining* yang berulang dan terintegrasi dalam sistem[43]. Dalam penelitian ini, *deployment* diwujudkan dalam bentuk aplikasi web sederhana berbasis Flask untuk klasifikasi judul buku

2.13 Google Colaboratory

Google Colaboratory, atau yang lebih dikenal dengan Google Colab, adalah layanan berbasis cloud yang memungkinkan pengguna untuk menulis dan menjalankan kode Python langsung melalui browser[44]. Platform ini sangat berguna untuk pengembangan dan eksperimen dalam bidang *data science*, *machine learning,* dan *deep learning.*

Salah satu keunggulan utama Google Colab adalah kemampuannya untuk menyediakan akses gratis ke GPU (*Graphics Processing Unit*), yang mempercepat proses pelatihan model dan analisis data besar. Google Colab mendukung berbagai pustaka Python populer seperti *TensorFlow*, *Keras*, dan *PyTorch*, sehingga memudahkan pengguna untuk membangun dan melatih model *machine learning*. Selain itu, pengguna dapat dengan mudah berbagi *notebook* dengan orang lain, memungkinkan kolaborasi yang efisien dalam proyek-proyek *data science*[44].

Dalam penelitian ini, Google Colab digunakan sebagai platform utama untuk pengembangan dan pelatihan model klasifikasi teks berbasis algoritma *Naive Bayes* dan LSTM. Kemampuan Google Colab dalam menangani dataset yang besar dan menyediakan lingkungan pemrograman interaktif sangat membantu dalam proses eksperimen. Selain itu, dukungan visualisasi langsung di *notebook* memungkinkan analisis data dan hasil evaluasi model dilakukan secara lebih efisien.

Dengan segala kelebihannya, Google Colab menjadi pilihan yang ideal untuk mendukung penelitian ini. Penggunaan Google Colab memungkinkan proses pengembangan model klasifikasi kategori buku menjadi lebih fleksibel, efisien, dan mudah diakses. Dukungan GPU gratis dan integrasi dengan pustaka machine learning membuat Google Colab sangat relevan untuk eksperimen berbasis data teks.

2.14 Python

Python adalah bahasa pemrograman tingkat tinggi yang populer karena sintaksnya yang mudah dipahami, fleksibilitas, dan dukungan komunitas yang besar[45]. Sejak pertama kali diperkenalkan pada tahun 1991 oleh Guido van Rossum, Python telah berkembang menjadi salah satu bahasa utama dalam bidang *data science*, *machine learning*, dan pengembangan perangkat lunak. Kemampuannya untuk menangani berbagai tugas, mulai dari analisis data hingga pembangunan model machine learning yang kompleks, membuat Python menjadi pilihan utama bagi banyak peneliti dan praktisi.

Salah satu keunggulan utama Python adalah kemudahannya untuk dipelajari[45]. Sintaks Python yang sederhana dan intuitif memungkinkan pengguna untuk

menulis kode dengan cepat dan efisien, bahkan bagi pemula sekalipun. Selain itu, Python juga dikenal karena fleksibilitasnya. Bahasa ini dapat digunakan untuk berbagai aplikasi, termasuk pengembangan web, otomatisasi, analisis data, dan machine learning. Fleksibilitas ini didukung oleh banyaknya library dan framework yang tersedia[45], seperti *NumPy* untuk komputasi numerik, *Pandas* untuk manipulasi data tabular, *Scikit-learn* untuk implementasi algoritma *machine learning* tradisional, serta *TensorFlow* dan *Keras* untuk pembangunan model *deep learning*.

2.15 Flask

Flask adalah sebuah micro web framework berbasis Python yang dirancang untuk memudahkan pengembangan aplikasi web dengan struktur yang ringan dan fleksibel[46]. Flask termasuk dalam kategori *minimalist framework*, karena tidak mengharuskan penggunaan struktur proyek tertentu dan tidak memiliki dependensi eksternal yang kompleks secara default, namun tetap mendukung perluasan fungsi dengan berbagai ekstensi seperti Flask-WTF, Flask-Login, atau Flask-RESTful.

Dalam konteks pengembangan aplikasi machine learning, Flask sangat populer digunakan untuk membangun antarmuka pengguna (user interface) sederhana atau layanan API (*Application Programming Interface*) yang memungkinkan model machine learning diakses oleh pengguna melalui browser atau aplikasi eksternal. Melalui Flask, model klasifikasi dapat dimuat, menerima input berupa judul buku dari pengguna, kemudian memproses input tersebut menggunakan model yang telah dilatih dan mengembalikan hasil prediksi kategori dalam format teks atau JSON.

Keunggulan Flask terletak pada kemudahannya dalam proses *deployment*[46], integrasi yang baik dengan pustaka *machine learning* seperti *Scikit-learn*, *TensorFlow*, dan *Pandas*, serta dokumentasi yang lengkap. Dengan menggunakan Flask, peneliti atau pengembang dapat membangun prototipe aplikasi web berbasis machine learning secara cepat dan efisien tanpa memerlukan arsitektur kompleks.

Dalam penelitian ini, Flask direncanakan digunakan untuk membangun antarmuka sederhana yang memungkinkan pengguna memasukkan judul buku dan

mendapatkan hasil klasifikasi kategori secara real-time. Penerapan Flask mendukung tahap deployment dalam siklus CRISP-DM, dengan tujuan akhir agar model klasifikasi yang telah dikembangkan dapat dimanfaatkan secara langsung oleh pustakawan maupun pengguna perpustakaan melalui sistem berbasis web.

2.16 Penelitian Terkait

Penelitian ini didasarkan pada berbagai studi terdahulu yang relevan dengan penerapan machine learning, khususnya algoritma *Naive Bayes* dan LSTM, dalam tugas klasifikasi teks. Studi-studi berikut memberikan landasan teoretis dan empiris yang memperkuat arah penelitian ini:

Studi komparatif dilakukan antara metode tradisional *machine learning* dan *deep* pada klasifikasi teks berita [2]. Studi ini menyoroti bahwa metode tradisional seperti *Naïve Bayes* dan SVM memiliki keunggulan dalam interpretabilitas tetapi mengalami kesulitan dalam menangani hubungan semantik yang kompleks dalam teks. Untuk mengatasi keterbatasan tersebut, penelitian ini mengusulkan model berbasis CNN, LSTM, dan *Attention Mechanism* guna meningkatkan akurasi klasifikasi. Dalam eksperimen yang dilakukan, penulis juga mengembangkan model klasifikasi berbasis *Naïve Bayes* dengan kombinasi TF-IDF sebagai metode ekstraksi fitur serta berbagai distribusi probabilitas *Naïve Bayes*. Hasilnya menunjukkan bahwa MNB dengan *TF-IDF* menghasilkan akurasi terbaik di antara varian *Naïve Bayes* lainnya, mencapai 88%. Namun, model *deep learning* berbasis *LSTM* masih lebih unggul dalam memahami konteks teks secara lebih mendalam dan mencapai akurasi yang lebih tinggi secara keseluruhan, yaitu 91%. Penelitian ini memberikan wawasan penting bahwa *Naïve Bayes* tetap relevan dalam klasifikasi teks karena efisiensi dan kecepatan pemrosesannya, tetapi *LSTM* lebih unggul dalam menangkap hubungan semantik yang lebih kompleks dalam teks. Oleh karena itu, studi ini memperkuat justifikasi penggunaan *Naïve Bayes* dan *LSTM* dalam penelitian ini, di mana keduanya akan dibandingkan dalam tugas klasifikasi kategori buku untuk mengevaluasi efektivitas masing-masing metode.

Algoritma *Multinomial Naïve Bayes* digunakan dalam penelitian [47] untuk mengklasifikasikan opini masyarakat di Twitter mengenai kebijakan *new normal*.

Studi ini menggunakan pendekatan CRISP-DM, yang mencakup tahap pemahaman bisnis, pemrosesan data, pemodelan, dan evaluasi model. Dalam eksperimen yang dilakukan, *Multinomial Naïve Bayes* digunakan untuk mengklasifikasikan sentimen positif dan negatif berdasarkan teks yang dikumpulkan melalui metode *web crawling*. Hasilnya menunjukkan bahwa algoritma ini mampu mengklasifikasikan sentimen dengan akurasi 90,25%, menjadikannya metode yang efisien dalam analisis teks skala besar. Penelitian ini juga mengidentifikasi tantangan dalam klasifikasi teks, seperti adanya kalimat yang mengandung sentimen campuran, yang dapat memengaruhi keakuratan model. Studi ini memberikan bukti bahwa MNB sangat cocok untuk tugas klasifikasi teks berbasis probabilistik dengan fitur seperti TF-IDF, sehingga memperkuat dasar penggunaan metode ini dalam penelitian ini untuk klasifikasi kategori buku di perpustakaan.

Eksplorasi penggunaan LSTM dalam klasifikasi sentimen teks dilakukan pada penelitian [48] terhadap komentar Weibo. Studi ini bertujuan untuk mengkategorikan sentimen pengguna ke dalam tiga kelas: positif, netral, dan negatif, dengan memanfaatkan kemampuan LSTM dalam menangkap dependensi jangka panjang dalam teks. Hasil eksperimen menunjukkan bahwa model LSTM mampu mencapai akurasi 98,31% dan F1-score 98,28%, jauh lebih tinggi dibandingkan metode tradisional seperti *Naïve Bayes* dan SVM. Keunggulan utama LSTM dalam penelitian ini adalah kemampuannya dalam memahami konteks kalimat yang panjang serta menangkap hubungan semantik yang lebih dalam dibandingkan model berbasis aturan atau probabilistik. Namun, studi ini juga menyoroti beberapa kelemahan LSTM, seperti kompleksitas komputasi yang tinggi dan performa yang menurun pada teks dengan ekspresi emosional kompleks seperti sarkasme atau humor. Oleh karena itu, penelitian ini menyarankan integrasi dengan model pra-terlatih atau peningkatan teknik ekstraksi fitur untuk meningkatkan akurasi lebih lanjut. Hasil dari studi ini memperkuat dasar penggunaan LSTM dalam penelitian ini, terutama dalam membandingkan performanya dengan *Naïve Bayes* dalam tugas klasifikasi kategori buku guna mengevaluasi efektivitas metode berbasis probabilistik dan *deep learning*.

Perbandingan performa antara *Multinomial Naïve Bayes* dan *Bernoulli Naïve Bayes* dilakukan dalam penelitian[49]. Studi ini bertujuan untuk menentukan model yang lebih optimal dalam mengklasifikasikan berita berdasarkan sentimen positif dan negatif. Dalam eksperimen yang dilakukan, MNB menunjukkan performa lebih baik karena mempertimbangkan frekuensi kata dalam dokumen, berbeda dengan *Bernoulli Naïve Bayes* yang hanya melihat keberadaan kata tanpa memperhitungkan jumlah kemunculannya. Hasil penelitian menunjukkan bahwa MNB mencapai akurasi 73,40%, lebih tinggi dibandingkan dengan *Bernoulli Naïve Bayes* yang hanya memperoleh 69%, mengindikasikan keunggulannya dalam klasifikasi teks berbasis kata. Studi ini memperkuat dasar penggunaan MNB dalam penelitian ini, terutama karena algoritma ini dikombinasikan dengan TF-IDF untuk menangani klasifikasi kategori buku secara lebih akurat berdasarkan frekuensi kata dalam judul buku.

Pendekatan jaringan LSTM berbasis memristor diusulkan dalam penelitian[50] untuk mengatasi tantangan konsumsi daya tinggi dan kompleksitas komputasi pada optimasi arsitektur. Dengan menggunakan *memristor crossbars*, model ini mampu mempercepat komputasi dan mengurangi kebutuhan daya tanpa mengorbankan akurasi secara signifikan. Eksperimen pada dataset IMDB movie reviews menunjukkan bahwa model ini mencapai akurasi 88,58%, dengan efisiensi komputasi yang lebih baik dibandingkan LSTM konvensional berbasis perangkat lunak. Studi ini relevan dengan penelitian ini karena menyoroti optimalisasi struktur LSTM dalam klasifikasi teks, yang sejalan dengan penggunaan *LSTM dengan custom word embeddings* dalam penelitian ini untuk klasifikasi kategori buku, sehingga teknik optimasi yang digunakan dapat menjadi referensi dalam meningkatkan efisiensi model yang dikembangkan.

Penerapan algoritma MNB untuk analisis sentimen juga dibahas dalam penelitian [51]. Studi ini berfokus pada klasifikasi ulasan film menggunakan model *Bag of Words* (BoW) dan TF-IDF sebagai metode ekstraksi fitur. Hasil eksperimen menunjukkan bahwa MNB mampu mencapai akurasi 91%, dengan peningkatan akurasi menjadi 93,1% setelah dioptimasi menggunakan regresi logistik. Penelitian ini menegaskan bahwa MNB tetap menjadi metode yang kuat dalam klasifikasi teks

karena kesederhanaan, kecepatan komputasi, serta kemampuannya dalam menangani teks skala besar. Studi ini sejalan dengan penelitian ini yang menggunakan MNB dengan TF-IDF untuk klasifikasi kategori buku, sehingga pendekatan yang digunakan dalam penelitian ini dapat menjadi referensi dalam pengolahan teks dan optimalisasi model klasifikasi.

Keunggulan LSTM dimanfaatkan dalam penelitian[52] untuk menganalisis sentimen destinasi wisata berbasis sawah dalam konteks ulasan pariwisata. Studi ini bertujuan untuk mengkategorikan sentimen positif dan negatif dalam empat aspek, yaitu fasilitas, layanan, kuliner, dan wahana, dengan data yang diperoleh secara otomatis dari *Google Maps Reviews*. Lima model LSTM dilatih berdasarkan kategori tersebut dan menghasilkan akurasi tertinggi sebesar 95,6% untuk kategori wahana serta rata-rata akurasi keseluruhan mencapai 91,48%. Penelitian ini menunjukkan bahwa LSTM mampu menangkap pola sentimen yang kompleks dalam teks dengan akurasi tinggi, terutama dalam tugas klasifikasi berbasis opini. Studi ini relevan dengan penelitian ini karena membuktikan efektivitas LSTM dalam mengolah teks secara lebih mendalam, sejalan dengan penggunaan LSTM dengan *word embeddings* dalam klasifikasi kategori buku yang dikembangkan dalam penelitian ini.

Selain pada teks bahasa alami, klasifikasi berbasis teks juga diterapkan pada kode pemrograman dalam penelitian[53] MNB. Studi ini berfokus pada penggunaan MNB untuk mengidentifikasi 12 bahasa pemrograman berdasarkan pola sintaksis dan struktur kode, dengan dataset yang mencakup 12.003 sampel kode sumber dari berbagai bahasa seperti Python, Java, C++, dan JavaScript. Hasil eksperimen menunjukkan bahwa model MNB mampu mencapai akurasi 95,09%, membuktikan efektivitasnya dalam klasifikasi berbasis teks dengan fitur probabilistik. Meskipun penelitian ini berfokus pada klasifikasi kode sumber, konsep penggunaan MNB dengan teknik ekstraksi fitur berbasis teks memiliki relevansi dengan penelitian ini dalam mengklasifikasikan kategori buku berdasarkan judul, sehingga pendekatan serupa dapat dijadikan referensi dalam pengolahan teks dan optimalisasi model klasifikasi.

Kemampuan LSTM dalam menangani data sekuensial yang kompleks dibuktikan dalam penelitian[54] melalui klasifikasi data *bandwidth* palsu, dengan tujuan meningkatkan deteksi manipulasi jaringan oleh penyedia layanan internet (ISP). Studi ini mengumpulkan 1.400 titik data bandwidth dari perangkat MikroTik RB 1100 AHx selama satu bulan, yang kemudian diproses dengan normalisasi dan dibagi menjadi 80% data pelatihan dan 20% data pengujian. Hasil eksperimen menunjukkan bahwa model LSTM mencapai akurasi 98,93%, membuktikan kemampuannya dalam membedakan antara bandwidth asli dan palsu dengan tingkat kesalahan klasifikasi hanya 1,07%. Studi ini menyoroti efektivitas LSTM dalam menangani data sekuensial yang kompleks serta potensinya dalam penerapan pemantauan jaringan secara *real-time*. Relevansi penelitian ini dengan skripsi ini terletak pada penerapan LSTM dalam tugas klasifikasi teks berbasis sekuensial, yang sejalan dengan penggunaan LSTM dengan *word embeddings* dalam klasifikasi kategori buku berdasarkan judul, sehingga strategi pemrosesan data dan optimalisasi model dalam studi ini dapat menjadi referensi yang berharga.

Terakhir, efektivitas kombinasi MNB dengan TF-IDF pada dataset IMDB dibahas dalam penelitian[55]. Studi ini membandingkan metode *unsupervised learning* seperti *Valence Aware Dictionary and Sentiment Reasoner* (VADER) dan *TextBlob* dengan metode *supervised learning* seperti *Bernoulli Naïve Bayes* dan MNB, di mana MNB dikombinasikan dengan TF-IDF untuk ekstraksi fitur. Hasil eksperimen menunjukkan bahwa kombinasi MNB dengan TF-IDF menghasilkan akurasi tertinggi sebesar 87,63%, mengungguli metode lainnya. Studi ini menunjukkan bahwa MNB tetap menjadi metode yang andal untuk klasifikasi teks dengan fitur berbasis probabilistik, terutama dalam tugas analisis sentimen. Relevansi penelitian ini terhadap skripsi ini terletak pada penggunaan MNB dengan TF-IDF untuk klasifikasi kategori buku, di mana pendekatan yang digunakan dalam studi ini dapat dijadikan referensi dalam pengolahan teks dan optimalisasi model klasifikasi.

Berbagai studi terdahulu yang telah dibahas menunjukkan bahwa *Naïve Bayes* dengan TF-IDF merupakan metode yang andal dalam klasifikasi teks karena efisiensinya dalam menangani data skala besar, sementara LSTM dengan *word embeddings* memiliki keunggulan dalam memahami hubungan semantik yang lebih

kompleks. Penelitian-penelitian sebelumnya juga menegaskan bahwa meskipun *Naïve Bayes* memiliki keterbatasan dalam menangkap konteks kata, teknik optimasi seperti seleksi fitur dan kombinasi dengan metode pembobotan dapat meningkatkan akurasinya. Di sisi lain, berbagai penelitian mengenai LSTM menunjukkan bahwa model ini unggul dalam menangani data teks sekuensial dengan akurasi tinggi, meskipun memiliki tantangan dalam efisiensi komputasi. Oleh karena itu, penelitian ini didasarkan pada perbandingan kedua metode tersebut dalam klasifikasi kategori buku, guna mengevaluasi efektivitas masing-masing algoritma dalam tugas klasifikasi berbasis teks serta menemukan solusi terbaik yang dapat diterapkan dalam sistem perpustakaan Universitas Lampung.

Tabel 2. 2 Penelitian Terkait

No	Penulis	Algoritma	Objek	Hasil
1.	Xiao Yu Li et al. (2024) [2]	*Multinomial Naïve Bayes* dengan *TF-IDF* dan LSTM	Teks Berita	Akurasi MNB: 88% Akurasi LSTM: 91%
2.	Zulfikar et al. (2023) [47]	*Multinomial Naïve Bayes*	Opini Twitter tentang Kebijakan *New Normal*	Akurasi: 90.25%
3.	Yin Qixuan (2024) [48]	LSTM	Komentar Weibo (Sentimen)	Akurasi: 98.31% F1-Score: 98.28%
4.	Gurinder Singh et al. (2019) [49]	*Multinomial Naïve Bayes*	Teks Berita (Sentimen)	Akurasi: 73.4%
5.	Gang Dou et al. (2023) [50]	LSTM berbasis *memristor*	Ulasan Film (IMDB *Movie Reviews*)	Akurasi: 88.58%
6.	Muhammad Abbas et al. (2019) [51]	*Multinomial Naïve Bayes*	Ulasan Film	Akurasi: 91%

Tabel 2. 2 (lanjutan)

7.	Njoto Benarkah et al. (2024) [52]	LSTM	Ulasan Wisata Sawah (*Google Maps*)	Akurasi tertinggi pada kategori wahana: 95.6% Rata-rata akurasi: 91.48%
8.	Ayman Hussein Odeh et al. (2023) [53]	*Multinomial Naïve Bayes*	Kode Sumber (*Source Code*) Bahasa Pemrograman	Akurasi: 95.09%
9.	Azriel Christian Nurcahyo et al. (2024) [54]	LSTM	Data *Bandwidth* Jaringan	Akurasi: 98.93%
10.	Christine Dewi et al. (2023) [55]	*Multinomial Naïve Bayes* dengan *TF-IDF*	Ulasan Film (Dataset IMDB)	Akurasi: 87.63%

Berdasarkan kajian terhadap penelitian-penelitian terdahulu yang dirangkum dalam tabel di atas, *State of the Art* penelitian ini terletak pada komparasi kinerja antara pengembangan metode probabilistik dan *deep learning* dalam menangani klasifikasi teks pendek berupa judul buku. Jika sebagian besar penelitian sebelumnya berfokus pada *Multinomial Naïve Bayes* standar atau LSTM standar, penelitian ini menerapkan varian algoritma yang lebih lanjut, yaitu Deep Feature Weighting Naïve Bayes (DFWNB). Metode ini menawarkan kebaruan dengan mengintegrasikan bobot fitur hasil seleksi statistik secara mendalam ke dalam estimasi probabilitas bersyarat untuk meningkatkan diskriminasi fitur . Di sisi lain, diterapkan pula Bidirectional LSTM (Bi-LSTM) yang diperkaya dengan GloVe Word Embedding untuk menangkap konteks semantik dua arah secara lebih komprehensif. Penelitian ini secara spesifik menguji ketahanan kedua model tersebut pada dataset nyata dari UPA Perpustakaan Universitas Lampung yang memiliki karakteristik ketidakseimbangan kelas (*imbalanced data*).

III. METODOLOGI PENELITIAN

3.1 Waktu dan Tempat

Penelitian ini dilaksanakan pada waktu dan tempat sebagai berikut:

1. Waktu Penelitian : Februari 2025 – Oktober 2025
2. Tempat penelitian : UPA Perpustakaan Universitas Lampung

Tabel 3. 1 Waktu Penelitian

No.	Kegiatan	Feb	Mar	Apr	Mei	Jun	Jul	Agu	Sep	Okt
1.	Studi Literatur									
2.	*Business Understanding*									
3.	*Data Understanding*									
4.	*Data Preparation*									
5.	*Modeling*									
6.	*Evaluation*									
7.	*Deployment*									
8.	Analisis									

3.2 Alat dan Bahan

Tabel 3. 2 Alat Penelitian

No.	Nama	Spesifikasi	Kegunaan
1.	Laptop	ACER Aspire 14 Intel 5-120U, RAM 16GB,	Hardware yang digunakan untuk membangun dan menguji model klasifikasi buku.

Tabel 3. 2 (lanjutan)

2.	Windows	Windows 11	Sistem Operasi.
3.	Python	Python 3	Bahasa pemrograman dalam membangun model machine learning.
4.	Google Colab	-	Layanan berbasis cloud yang digunakan untuk menulis dan menjalankan kode Python.
5.	Vscode		Untuk implementasi ke dalam sistem
6.	Pustaka (Libraries	*Pandas* dan *NumPy*	Untuk manipulasi data
		Scikit-learn	Untuk preprocessing dan membangun model Naïve Bayes
		TensorFlow dan *Keras*	Untuk membangun model Deep Learning LSTM
		Flask	Untuk pembuatan *web deployment*
		Matplotlib dan *Seaborn*	Untuk visualisasi data

Adapun untuk bahannya adalah sebagai berikut:

1. **Dataset Judul Buku:** Data buku yang didapatkan dari layanan cadangan UPA Perpustakaan Universitas Lampung. Perpustakaan memiliki koleksi judul buku dengan total 161.529 eksemplar. Atribut yang digunakan difokuskan pada "Judul Buku" sebagai fitur teks dan "Nomor Klasifikasi DDC" sebagai label kategori.

Tabel 3. 3 Sampel Data (30/161.529)

No	Judul Buku	Kategori
1	pedoman praktis membuat usulan penelitian masri singarimbun	000
2	research games an aprroach to the study of decision processes k c bowen	000
3	pemograman komputer ftk 116	000
4	filsafat publik analisis seorang pemikir politik terkemuka atas tantangan yang dihadapi negara	100
5	praktik transparansi dialog menurut para filsuf sebuah refleksi praktis filosofi dany harianto	100
6	philosophy paradox and discovery arthur j minton	100
7	what happens if i say damn you god charles victor arokiasamy	200
8	moderasi beragama lukman hakim saifuddin et al	200
9	pendidikan agama islam arah baru pengembangan ilmu dan kepribadian di perguruan tinggi	200
10	ilmu sosial dasar mata kulian dasar umum	300
11	metode analisis data sosial basrowi sunyono	300
12	hukum acara penyelesaian perselisihan hubungan industrial tata cara dan proses penyelesaian sen	300
13	bahasa indonesia anda bertanya inilah jawabnya	400
14	kamus ungkapan indonesia inggris pandai berbahasa inggris	400
15	the modern reader s japanese english character dictionary	400
16	macmillan reviced encyclopedia of science matter and energy	500
17	linear algebra based on schaum s outline of theory and problems of linear algebra 3rd ed seymou	500
18	entomologi pertanian jumar	500
19	educational technology a definition with commentary alan januszewski michael molenda	600
20	mekanisme dan dinamika mesin ramses y hutahaean	600
21	dasar dasar teknik mesin daryanto	600
22	the modern art invasion picasso duchamp and the 1913 armory show that scandalized america eliza	700
23	batik nusantara makna filosofis cara pembuatan industri batik edisi 1 ari wulandari	700
24	peran pemuda dalam kebangkitan film indonesia misbach yusa biran	700
25	apresiasi sastra indonesia moha junaedie	800
26	merpati biru ed 2 achmad munif	800

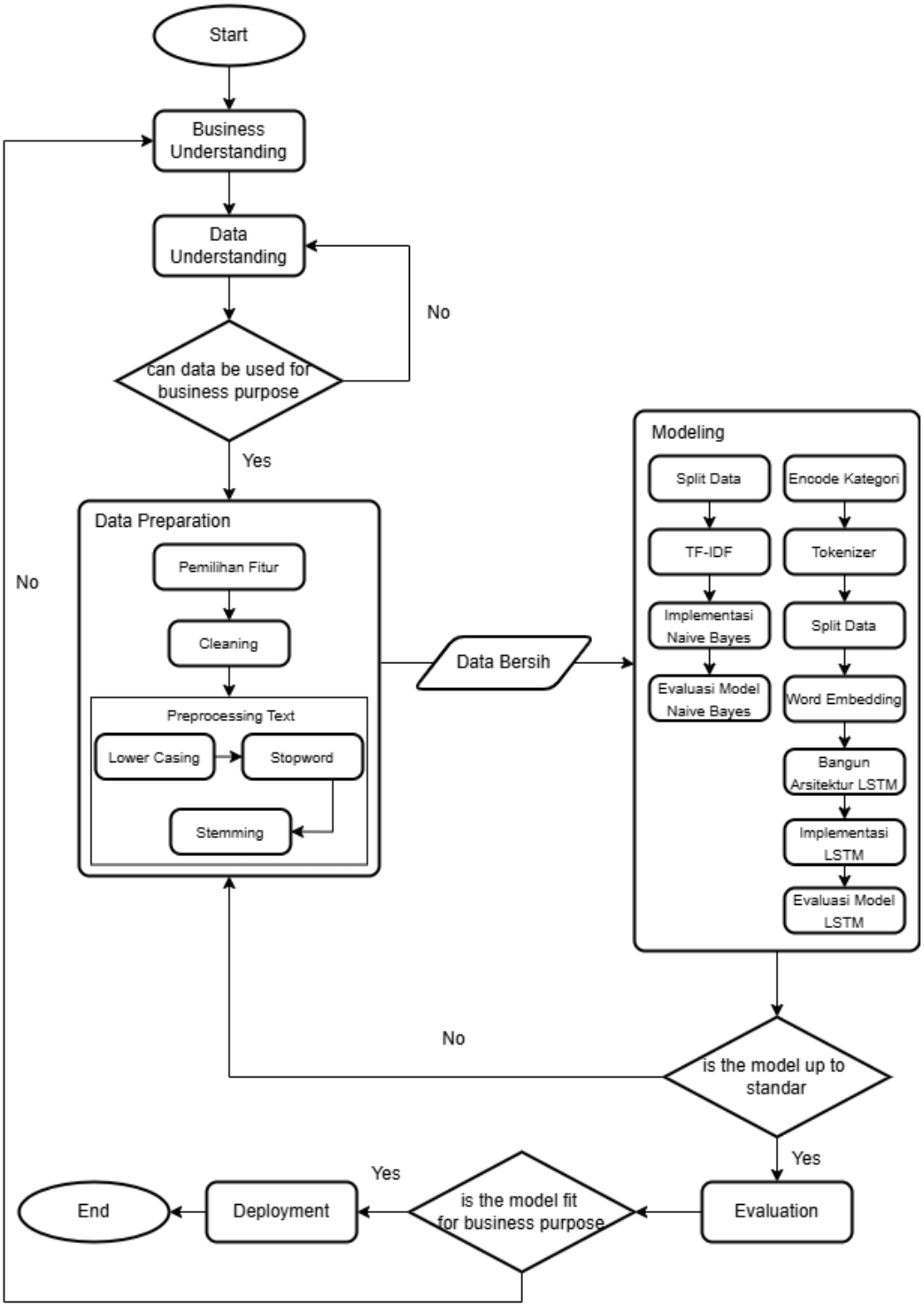
Tabel 3.3 (lanjutan)

27	babad tanah jawi mitologi legenda folklor dan kisah raja raja jawa buku ii penerjemah amir roch	800
28	the state of indians environment 1984 1985 the second citizens report	900
29	dua kota tiga zaman surabaya dan malang sejak kolonial sampai kemerdekaan purnawan basundoro	900
30	the united states an interpretive history r kent fielding	900

2. *Pre-trained Word Embedding* GloVe. Data pendukung berupa vektor representasi kata yang telah dilatih sebelumnya (*pre-trained*). Penelitian ini menggunakan GloVe varian glove.6B.300d.txt yang dikembangkan oleh Stanford. File ini berisi representasi vektor 300 dimensi untuk 400.000 kosakata umum. Bahan ini digunakan sebagai inisialisasi bobot pada lapisan *embedding* model LSTM untuk menangkap hubungan semantik antar kata[39].
3. *Pre-trained Word Embedding* FastText. File vektor kata pra-latih FastText edisi cc.id.300.vec yang dikembangkan oleh *Facebook AI Research* (FAIR). Model ini dilatih menggunakan metode *Continuous Bag-of-Words* (CBOW) pada korpus *Common Crawl* dan Wikipedia Bahasa Indonesia[40].

3.3 Tahapan Penelitian

Penelitian ini dilakukan melalui beberapa tahapan sistematis untuk mencapai tujuan yang telah ditetapkan. Setiap tahapan dirancang agar proses pengembangan model klasifikasi kategori buku berbasis algoritma *Naïve Bayes* dan LSTM dapat berjalan secara optimal.



Gambar 3. 1 Flowchart Penelitian

Berdasarkan metodologi CRISP-DM, tahapan yang dilakukan dalam penelitian ini mencakup enam tahap utama sebagai berikut:

3.3.1 *Business Understanding*

Tahap ini diawali dengan identifikasi permasalahan yang terjadi di UPA Perpustakaan Universitas Lampung, yaitu kesulitan dalam menentukan kategori buku secara manual akibat migrasi sistem yang telah terjadi beberapa kali (*Dynix, SLiMS, eLib*, dan saat ini dalam proses migrasi ke *INLISLite*). Migrasi ini menyebabkan hilangnya sebagian informasi kategori buku. Akibatnya pustakawan kesulitan dalam menentukan kembali data kategori buku yang hilang tersebut serta mengklasifikasikan buku yang baru. Dalam proses katalogisasi buku sendiri penentuan kategori ini merupakan tahap awal, yaitu pustakawan melakukan identifikasi buku yang meliputi judul, tahun terbit, kategori, dan lain-lain.

Proses identifikasi masalah dilakukan melalui diskusi dan wawancara dengan pustakawan, serta analisis terhadap sistem manajemen perpustakaan yang ada. Hasil analisis menunjukkan bahwa metode klasifikasi manual saat ini memerlukan waktu dan tenaga yang besar, serta rentan terhadap inkonsistensi dan kesalahan penentuan kategori. Oleh karena itu, solusi yang diusulkan dalam penelitian ini adalah mengembangkan model klasifikasi otomatis dengan memanfaatkan teknologi *machine learning*, yaitu menggunakan algoritma *Naïve Bayes* dengan TF-IDF dan LSTM dengan *word embeddings*. Model ini diharapkan dapat meningkatkan efisiensi serta akurasi dalam pengelolaan kategori buku, sehingga mempermudah pustakawan dan pengguna perpustakaan dalam mengakses informasi koleksi.

Output dari tahap ini adalah pemahaman yang jelas tentang permasalahan yang dihadapi, kebutuhan sistem klasifikasi otomatis, serta penentuan tujuan penelitian, yaitu pengembangan model klasifikasi kategori buku berbasis *machine learning*.

3.3.2 *Data Understanding*

Tahap ini bertujuan untuk memahami karakteristik data yang digunakan dalam penelitian guna memastikan kesesuaiannya dengan tujuan klasifikasi kategori buku. Data yang digunakan berupa judul buku beserta kategorinya, yang diperoleh dari layanan cadangan perpustakaan Universitas Lampung. Data diekstrak dalam format yang dapat diolah lebih lanjut, seperti *CSV* atau *Excel*, kemudian diperiksa untuk

memastikan kelengkapan informasi, khususnya terkait judul buku dan kategorinya. Jika ditemukan kategori yang hilang atau tidak sesuai, data akan ditandai untuk diproses lebih lanjut, sementara validasi dilakukan dengan membandingkan data yang dikumpulkan dengan informasi dari katalog perpustakaan guna memastikan konsistensinya.

Setelah data terkumpul, dilakukan eksplorasi untuk menganalisis distribusi kategori buku serta mengidentifikasi potensi permasalahan seperti data yang hilang, kategori yang tidak konsisten, atau duplikasi data yang nantinya akan diproses di *data preparation*. Dataset yang digunakan dalam penelitian ini sekitar 161.529 eksemplar data buku dengan 10 kategori/kelas.

Output dari tahap ini adalah dataset yang telah terkumpul dalam bentuk *Excel* dan terverifikasi serta pemahaman yang lebih mendalam mengenai struktur data, termasuk distribusi kategori dan potensi permasalahan yang harus diselesaikan sebelum masuk ke tahap *data preparation*.

3.3.3 *Data Preparation*

Tahap ini berfokus pada pemrosesan data teks agar dapat diolah secara optimal. Proses ini dimulai dengan pemilihan fitur, cleaning data dan dilanjutkan dengan proses *pre-processing*. Tahapan *pre-processing* mencakup *lower casing*, penghapusan *stopwords*, dan *stemming*. Adapun library yang digunakan untuk tahapan ini adalah library *TensorFlow/Keras* dan *Scikit-learn*.

Output dari tahap ini adalah dataset yang telah diproses dan siap digunakan untuk pelatihan model, dengan teks yang telah dibersihkan serta melalui tahapan preprocessing teks.

3.3.3.1 Pemilihan Fitur

Tahap ini bertujuan untuk menyeleksi atribut data yang relevan dari dataset mentah perpustakaan. Dari sekian banyak kolom yang tersedia, penelitian ini hanya akan menggunakan dua atribut utama, yaitu judul buku sebagai fitur input (*input feature*) dan nomor klasifikasi DDC sebagai label target. Jika terdapat pemisahan antara judul utama dan anak judul, kedua kolom tersebut akan digabungkan (*concatenate*)

menjadi satu kesatuan teks judul yang utuh untuk memaksimalkan informasi konteks yang diterima oleh model.

3.3.3.2 Cleaning Data

Pembersihan data dilakukan untuk memastikan kualitas dataset sebelum masuk ke tahap pemrosesan lebih lanjut. Langkah-langkah yang dilakukan meliputi:

a. Penghapusan Duplikasi: Mengeliminasi data judul buku yang ganda (*duplicate*) agar tidak terjadi bias pada saat pelatihan model.
b. Penanganan *Missing Values*: Menghapus baris data yang memiliki nilai kosong (*null/NaN*), baik pada kolom judul maupun kategori.
c. Standardisasi Label Kategori: Menyesuaikan format nomor kelas DDC menjadi 10 kategori utama (kelas besar). Nomor klasifikasi yang spesifik akan dibulatkan ke bawah menjadi kelipatan 100 (misalnya, 621 menjadi 600) dan diformat menjadi *string* tiga digit (misalnya, 0 menjadi '000') agar sesuai dengan standar 10 kelas utama DDC.

3.3.3.3 *Pre-processing Text*

Pre-processing meliputi beberapa tahapan, antara lain:

3.3.3.3.1 *Lower Casing*

Lower casing adalah proses mengubah semua huruf dalam teks menjadi huruf kecil, yaitu kata-kata yang terdapat pada judul buku. Tahap ini penting untuk memastikan konsistensi dalam pemrosesan teks, karena kata-kata seperti "Python" dan "python" akan dianggap sama oleh model. Dengan melakukan *lower casing*, mengurangi kompleksitas data dan menghindari duplikasi kata yang sebenarnya sama tetapi berbeda dalam penulisan huruf besar-kecil.

Tabel 3. 4 *Lower Casing*

No	Sebelum	Sesudah
1	Pemrograman Komputer Fltk 116	pemrograman komputer fltk 116
2	Filsafat Publik Analisis Seorang Pemikir Politik Terkemuka Atas Tantangan Yang Dihadapi Negara	filsafat publik analisis seorang pemikir politik terkemuka atas tantangan yang dihadapi negara

Tabel 3. 4 (lanjutan)

3	Pendidikan Agama Islam Arab Baru Pengembangan Ilmu Dan Kepribadian Di Perguruan Tinggi	pendidikan agama islam arab baru pengembangan ilmu dan kepribadian di perguruan tinggi
4	Hukum Acara Penyelesaian Perselisihan Hubungan Industrial Tata Bahasa Indonesia	hukum acara penyelesaian perselisihan hubungan industrial tata bahasa indonesia
5	The Modern Art Invasion Picasso Duchamp And The 1913 Armory	the modern art invasion picasso duchamp and the 1913 armory

3.3.3.3.2 Penghapusan *stopword*

Penghapusan *stopword* adalah proses menghilangkan kata-kata yang sering muncul tetapi tidak memiliki makna penting dalam judul buku, seperti "dan", "di", "untuk", atau "yang". Kata-kata ini dihilangkan karena tidak memberikan kontribusi signifikan terhadap pemahaman makna teks. Dengan menghapus *stopword*, mengurangi dimensi data dan fokus pada kata-kata yang lebih relevan.

Tabel 3. 5 Penghapusan *Stopword*

No	Sebelum	Sesudah
1	['pemrograman', 'komputer', 'fltk', '116']	['pemrograman', 'komputer', 'fltk', '116']
2	['filsafat', 'publik', 'analisis', 'seorang', 'pemikir', 'politik', 'terkemuka', 'atas', 'tantangan', 'yang', 'dihadapi', 'negara']	['filsafat', 'publik', 'analisis', 'pemikir', 'politik', 'terkemuka', 'tantangan', 'dihadapi', 'negara']
3	['pendidikan', 'agama', 'islam', 'arab', 'baru', 'pengembangan', 'ilmu', 'dan', 'kepribadian', 'di', 'perguruan', 'tinggi']	['pendidikan', 'agama', 'islam', 'arab', 'pengembangan', 'ilmu', 'kepribadian', 'perguruan', 'tinggi']
4	['hukum', 'acara', 'penyelesaian', 'perselisihan', 'hubungan', 'industrial', 'tata', 'bahasa', 'indonesia']	['hukum', 'acara', 'penyelesaian', 'perselisihan', 'hubungan', 'industrial', 'tata', 'bahasa', 'indonesia']
5	['the', 'modern', 'art', 'invasion', 'picasso', 'duchamp', 'and', 'the', '1913', 'armory']	['modern', 'art', 'invasion', 'picasso', 'duchamp', '1913', 'armory']

3.3.3.3.3 Stemming

Stemming adalah proses mengubah kata ke bentuk kata dasarnya dengan menghilangkan imbuhan atau variasi kata. Tujuan *stemming* adalah untuk mengurangi variasi kata yang sebenarnya memiliki makna serupa, sehingga model dapat lebih efisien dalam memproses teks. Meskipun *stemming* terkadang menghasilkan kata yang tidak sempurna secara linguistik, metode ini tetap berguna untuk menyederhanakan data teks.

Tabel 3. 6 *Stemming*

No	Sebelum	Sesudah
1	['pemrograman', 'komputer', 'fltk', '116']	['program', 'komputer', 'fltk', '116']
2	['filsafat', 'publik', 'analisis', 'pemikir', 'politik', 'terkemuka', 'tantangan', 'dihadapi', 'negara']	['filsafat', 'publik', 'analisis', 'pikir', 'politik', 'terkemuk', 'tantang', 'hadap', 'negara']
3	['pendidikan', 'agama', 'islam', 'arab', 'pengembangan', 'ilmu', 'kepribadian', 'perguruan', 'tinggi']	['didik', 'agama', 'islam', 'arab', 'kembang', 'ilmu', 'pribadi', 'guruan', 'tinggi']
4	['hukum', 'acara', 'penyelesaian', 'perselisihan', 'hubungan', 'industrial', 'tata', 'bahasa', 'indonesia']	['hukum', 'acara', 'selesai', 'selisih', 'hubung', 'industri', 'tata', 'bahasa', 'indonesia']
5	['modern', 'art', 'invasion', 'picasso', 'duchamp', '1913', 'armory']	['modern', 'art', 'invasi', 'picasso', 'duchamp', '1913', 'armory']

3.3.4 *Modeling*

Tahap ini berfokus pada pengembangan model klasifikasi kategori buku menggunakan dua algoritma, *Naïve Bayes* dan LSTM. Model *Naïve Bayes* dikembangkan menggunakan library *Scikit-learn*, dengan data yang telah diproses menggunakan TF-IDF sebagai representasi fitur. Model ini bekerja dengan mempelajari hubungan antara frekuensi kemunculan kata dalam judul buku dan kategori yang relevan berdasarkan pendekatan probabilistik.

Di sisi lain, model LSTM dikembangkan menggunakan library *TensorFlow/Keras*, dengan data yang nanti direpresentasikan dengan pre-trained *word embedding*. Proses ini memungkinkan model untuk memahami hubungan semantik antar kata

berdasarkan konteks penggunaannya dalam dataset. Model LSTM dirancang untuk menangkap pola sekuensial dalam judul buku, yang memungkinkan model untuk memprediksi kategori dengan mempertimbangkan keterkaitan antar kata dalam suatu judul.

Output dari tahap ini adalah dua model klasifikasi yang telah dilatih. Setelah model dilatih, dilakukan pengecekan awal untuk memastikan kualitas model sebelum masuk ke tahap evaluasi final. Standar kelayakan model yang ditetapkan dalam penelitian ini meliputi tiga kriteria utama:

1. Ambang Batas Akurasi (*Accuracy Threshold*): Model dinyatakan layak jika mampu menghasilkan akurasi validasi minimal 60%. Penentuan angka ini mengacu pada standar interpretasi nilai kesepakatan (*agreement*) yang dikemukakan oleh Landis & Koch (1977). Dalam standar tersebut, rentang nilai 0.61 – 0.80 (61% - 80%) dikategorikan sebagai "Substantial" (Kuat). Oleh karena itu, ambang batas 60% dipilih sebagai indikator minimum bahwa model telah memiliki kemampuan prediksi yang substansial dan jauh melampaui probabilitas tebakan acak (*random guessing*)[56].
2. Tidak terjadi *Overfitting* atau *Underfitting* Ekstrem**:** Selisih (*gap*) antara akurasi data latih (*training*) dan data validasi (*validation*) pada model LSTM dibatasi maksimal 10%. Model yang baik harus menyeimbangkan antara kinerja pada data latih dan kemampuan generalisasi pada data baru. Selisih performa yang melebihi 10% mengindikasikan bahwa model memiliki *high variance*, di mana model terlalu spesifik mempelajari *noise* pada data latih dan gagal melakukan generalisasi, sehingga pelatihan perlu dihentikan atau diregulasi ulang.
3. Konvergensi Fungsi Loss: Grafik fungsi *loss* pada data latih dan validasi harus menunjukkan tren penurunan yang stabil seiring bertambahnya *epoch* dan mencapai titik konvergen (tidak lagi mengalami fluktuasi signifikan). Hal ini menandakan bahwa algoritma optimisasi telah berhasil meminimalkan kesalahan prediksi model secara efektif.

Jika model tidak memenuhi standar tersebut, maka proses akan dikembalikan ke tahap *ata Preparation* untuk dilakukan peninjauan ulang.

3.3.4.1 *Modeling Naïve Bayes*

3.3.4.1.1 Pemisahan Data

Untuk pemodelan *Naïve Bayes*, data dibagi menjadi dua bagian, yaitu data latih (*training set*) sebesar 80% dan data uji (*testing set*) sebesar 20%. Proses pemisahan ini dilakukan menggunakan teknik *stratified sampling* guna memastikan proporsi setiap kategori buku tetap konsisten dan terwakili secara merata, baik dalam data latih maupun data uji.

3.3.4.1.2 Representasi Teks dengan TF-IDF

TF-IDF merupakan teknik pembobotan kata yang digunakan untuk modeling *Naive Bayes.* Disini dicontohkan perhitungan manual TF-IDF dengan memakai 15 kata sebagai berikut:

['program', 'komputer', 'fltk', '116', 'filsafat', 'publik', 'analisis', 'pikir', 'politik', 'agama', 'islam', 'arab', 'ilmu', 'bahasa', 'indonesia']

Selanjutnya dihitung *term frequency* untuk setiap kata, disini dicontohkan dengan 5 kategori, perhitungan dilakukan dengan persamaan (16):

Tabel 3. 7 *Term Frequency* (TF)

Kata	Kategori 000	Kategori 100	Kategori 200	Kategori 300	Kategori 400
program	1/20 = 0.05	0	0	0	0
komputer	1/20 = 0.05	0	0	0	0
fltk	1/20 = 0.05	0	0	0	0
116	1/20 = 0.05	0	0	0	0
filsafat	0	2/20 = 0.01	0	0	0
publik	0	1/20 = 0.05	0	0	0
analisis	0	1/20 = 0.05	0	0	0
pikir	0	1/20 = 0.05	0	0	0
politik	0	1/20 = 0.05	0	0	0

Tabel 3. 7 (lanjutan)

agama	0	0	1/22 = 0.045	0	0
islam	0	0	2/22 = 0.09	0	0
arab	0	0	1/22 = 0.045	0	0
ilmu	0	0	1/22 = 0.045	1/20 = 0.05	0
bahasa	0	0	0	0	2/18 = 0.11
indonesia	0	0	0	0	2/18 = 0.11

Lanjut ke hitung *Inverse Document Frequency* (IDF) dengan persamaan (17):

Tabel 3. 8 *Inverse Document Frequency* (IDF)

Kata	**DF (w)**	**IDF**
program	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
komputer	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
fltk	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
116	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
filsafat	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
publik	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
analisis	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
pikir	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
politik	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
agama	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
islam	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
arab	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
ilmu	2	$\log\frac{5}{1+2} = \log\frac{5}{3} = 0.222$
bahasa	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$
indonesia	1	$\log\frac{5}{1+1} = \log 2.5 = 0.398$

Setelah itu lanjut ke perhitungan TF-IDF dengan persamaan (18):

Tabel 3. 9 TF-IDF

Kata	Kategori 000	Kategori 100	Kategori 200	Kategori 300	Kategori 400
program	0.05 x 0.398 = 0.199				
komputer	0.05 x 0.398 = 0.199				
fltk	0.05 x 0.398 = 0.199				
116	0.05 x 0.398 = 0.199				
filsafat		0.1 x 0.398 = 0.004			
publik		0.05 x 0.398 = 0.199			
analisis		0.05 x 0.398 = 0.199			
pikir		0.05 x 0.398 = 0.199			
politik		0.05 x 0.398 = 0.199			
agama			0.045 x 0.398 = 0.0179		
islam			0.09 x 0.398 = 0.0358		
arab			0.045 x 0.398 = 0.0179		
ilmu			0.045 x 0.222 = 0.00999	0.05 x 0.222 = 0.0111	
bahasa					0.11 x 0.398 = 0.0438
indonesia					0.11 x 0.398 = 0.0438

Nilai dengan TF-IDF tinggi menunjukkan kata lebih unik dan penting di kategori itu.

3.3.4.1.3 Implementasi *Naïve Bayes*

Dalam *modeling* menggunakan *Naïve Bayes* diawali dengan menghitung probabilitas *prior* dengan rumus sebagai berikut:

$$P(kategori) = \frac{Jumlah\ kata\ di\ kategori}{total\ jumlah\ kata} \tag{25}$$

Dari data sebelumnya, jumlah kata per kategori:

- Kategori 000: 20 kata
- Kategori 100: 20 kata
- Kategori 200: 22 kata
- Kategori 300: 20 kata
- Kategori 400: 18 kata

Total semua kata = 100 kata

Lalu hitung probilitas *prior* untuk tiap kategori:

- P(Kategori 000) = 20/100 = 0.2
- P(Kategori 100) = 20/100 = 0.2
- P(Kategori 200) = 22/100 = 0.22
- P(Kategori 300) = 20/100 = 0.2
- P(Kategori 400) = 18/100 = 0.18

Setelah itu hitung probabilitas *likelihood* dengan rumus:

Contoh perhitungan:

$$P(\text{"}program\text{"} \mid Kategori\ 000) = \frac{1 + 1}{20 + 100} = 0.0167 \tag{26}$$

Tabel 3. 10 Probabilitas *Likelihood*

Kata	Kategori 000	Kategori 100	Kategori 200	Kategori 300	Kate-gori 400
program	0.0167	0.0083	0.0082	0.0083	0.0085
komputer	0.0167	0.0083	0.0082	0.0083	0.0085
filsafat	0.01	0.0167	0.0082	0.0083	0.0085
agama	0.0083	0.0083	0.0164	0.0083	0.0085
islam	0.0083	0.0083	0.0246	0.0083	0.0085
ilmu	0.0083	0.0083	0.0082	0.0167	0.0085
bahasa	0.0083	0.0083	0.0082	0.0083	0.0169

Nilai probabilitas *prior* dan *likelihood* ini nantinya yang akan dipakai untuk menentukan kategori suatu judul buku. Contoh ada buku baru dengan judul "*Analisis filsafat politik dalam pendidikan agama islam*". Kata-kata yang akan masuk ke dalam vocabulary: ['analisis', 'filsafat', 'politik', 'pendidikan', 'agama', 'islam'].

Berdasarkan persamaan utama *Naïve Bayes* (1) artinya nilai probabilitas *prior* dikalikan dengan semua *likelihood* kata-kata dalam kategori tersebut.

- P(000|dokumen) = $0.2\times(0.0083\times0.0083\times0.0083\times0.0083\times0.0083\times0.0083)$ = $0.2\times3.61\times10-10 = 7.22\times10^{-11}$
- P(100|dokumen) = $0.2\times(0.057\times0.0167\times0.057\times0.0083\times0.0083\times0.0083)$ = $0.2\times2.16\times10-6 = 4.32\times10^{-7}$
- P(200|dokumen) = $0.22\times(0.0082\times0.0082\times0.0082\times0.0082\times0.054\times0.0246)$ = $0.22\times3.97\times10-7 \approx 8.73\times10^{-8}$
- P(300|dokumen) = $0.2\times(0.0083\times0.0083\times0.0083\times0.0083\times0.0083\times0.0083)$ = $0.2\times3.61\times10-10 = 7.22\times10^{-11}$
- (400|dokumen) = $0.18\times(0.0085\times0.0085\times0.0085\times0.0085\times0.0085\times0.0085)$ = $0.18\times3.87\times10-10 = 6.96\times10^{-11}$

Kategori dengan probabilitas terbesar adalah Kategori 100, maka Judul buku "*Analisis filsafat politik dalam pendidikan agama islam*" kemungkinan besar masuk ke kategori Filsafat dan Psikologi.

3.3.4.1.4 Evaluasi Model Naïve Bayes

Tahap evaluasi ini bertujuan untuk mengukur kinerja model *Naïve Bayes* dan membandingkannya dengan beberapa varian algoritma *Naïve Bayes*. Perbandingan ini dilakukan untuk memilih model *Naive Bayes* yang terbaik. Evaluasi dilakukan menggunakan data uji dengan mengukur lima indikator kinerja utama, yaitu: akurasi, presisi, *recall*, *F1-Score*, dan waktu pelatihan. Kinerja model diukur menggunakan *Confusion Matrix* untuk melihat distribusi prediksi benar dan salah pada setiap kelas.

3.3.4.2 *Modeling* LSTM

3.3.4.2.1 *Encoding* Kategori

Langkah pertama dalam persiapan data untuk model LSTM adalah transformasi label target. Label kategori buku yang pada awalnya berupa data nominal atau kode teks (misalnya '000', '100', dst.) dikonversi menjadi nilai numerik (integer) menggunakan teknik *Label Encoding*. Proses ini memetakan setiap kategori unik ke dalam angka bulat mulai dari 0 hingga N-1 (dalam hal ini 0 hingga 9 untuk 10 kelas kategori DDC). Konversi ini mutlak diperlukan agar variabel target dapat diproses secara matematis oleh algoritma jaringan saraf,

3.3.4.2.2 Tokenizer

Tokenizer adalah proses pemisahan antar kata dalam setiap judul buku. Dalam. Tahap ini penting karena model memerlukan input dalam bentuk token untuk dapat memproses dan menganalisis teks.

Tabel 3. 11 *Tokeziner*

No	Sebelum	Sesudah
1	pemrograman komputer fltk 116	['pemrograman', 'komputer', 'fltk', '116']
2	filsafat publik analisis seorang pemikir politik terkemuka atas tantangan yang dihadapi negara	['filsafat', 'publik', 'analisis', 'seorang', 'pemikir', 'politik', 'terkemuka', 'atas', 'tantangan', 'yang', 'dihadapi', 'negara']
3	pendidikan agama islam arab baru pengembangan ilmu dan kepribadian di perguruan tinggi	['pendidikan', 'agama', 'islam', 'arab', 'baru', 'pengembangan', 'ilmu', 'dan', 'kepribadian', 'di', 'perguruan', 'ting-gi']
4	hukum acara penyelesaian perse-lisihan hubungan industrial tata bahasa indonesia	['hukum', 'acara', 'penyelesaian', 'perselisihan', 'hubungan', 'industrial', 'tata', 'bahasa', 'indonesia']
5	the modern art invasion picasso duchamp and the 1913 armory	['the', 'modern', 'art', 'invasion', 'pi-casso', 'duchamp', 'and', 'the', '1913', 'armory']

3.3.4.2.3 Pemisahan Data

Dataset dibagi menjadi tiga dengan proporsi 70:20:10 menggunakan teknik *stratified sampling*. Teknik ini memastikan distribusi kategori buku pada setiap bagian data tetap proporsional sesuai dengan data aslinya. Pembagiannya data latih (*training set*) 70%, data validasi (*validation set*) 20%, dan data uji (*testing set*) 10%.

3.3.4.2.4 Word Embedding

Word embedding adalah teknik representasi teks yang akan digunakan untuk LSTM untuk mengubah kata-kata menjadi vektor numerik berdimensi tetap, di mana vektor ini menangkap makna semantik dan hubungan antar kata.

Misalnya, ditentukan vektor untuk tiap kata secara acak terkontrol atau belajar dari data nanti sebagai berikut:

Tabel 3. 12 *Word Embedding*

Kata	Vektor Embedding (3 dimensi)
program	[0.4, 0.2, 0.1]
komputer	[0.6, 0.3, 0.2]
fltk	[0.1, 0.7, 0.4]
116	[0.2, 0.5, 0.8]
filsafat	[0.7, 0.3, 0.6]
publik	[0.3, 0.8, 0.5]
analisis	[0.5, 0.4, 0.7]
pikir	[0.6, 0.6, 0.3]
politik	[0.4, 0.5, 0.5]
agama	[0.8, 0.2, 0.6]
islam	[0.9, 0.3, 0.7]
arab	[0.7, 0.5, 0.4]
ilmu	[0.4, 0.7, 0.2]
bahasa	[0.3, 0.6, 0.8]
indonesia	[0.5, 0.4, 0.6]

Nantinya kata yang mirip maknanya akan mempunyai pola vektor yang agak mirip. Dalam *word embedding* ini nantinya data akan belajar sendiri, oleh karena itu semakin banyak datanya maka akan semakin bagus

3.3.4.2.5 Arsitektur Model LSTM

Pada tahap ini, dilakukan perancangan arsitektur LSTM berbasis model sekuensial untuk menangani tugas klasifikasi teks. Desain arsitektur yang diusulkan bertujuan untuk memaksimalkan kemampuan model dalam memahami konteks semantik dan sintaksis dari judul buku.

3.3.4.2.6 Pelatihan Model

Dalam *modeling* ini, LSTM mengingat atau memproses judul buku berdasarkan urutan kata, berbeda dengan *Naïve Bayes* yang mengasumsikan setiap kata independen dengan yang lain.

Misal ada judul buku baru "Analisis filsafat politik dalam pendidikan agama islam", setelah di *pre-processing*: [′analisis′,′filsafat′,′politik′,′pendidikan′,′agama′,′islam′]

Kata-kata tersebut diubah menjadi vektor embedding:

[0.5,0.4,0.7],[0.7,0.3,0.6],[0.4,0.5,0.5],[0.8,0.2,0.6],[0.9,0.3,0.7]]

Untuk masuk ke LSTM semuanya harus memiliki panjang yang sama yang nantinya akan diberikan ketentuan panjangnya pada arsitektur LSTM mengikuti judul terpanjang, jadi kalau ada judul yang pendek ditambahkan *padding* (0), jadi:

[[0.5,0.4,0.7],[0.7,0.3,0.6],[0.4,0.5,0.5],[0.8,0.2,0.6],[0.9,0.3,0.7],[0,0,0],[0,0,0]]

Ini akan menjadi input shape:

- 7 timesteps (panjang kalimat setelah *padding*).
- 3 fitur per *timestep* (embedding 3 dimensi).

Timestep pertama:

- Input: [0.5, 0.4, 0.7]
- *Hidden state* awal: [0, 0, 0]
- *Cell state* awal: [0, 0, 0]

a. Forget Gate

Perhitungan *forget gate* dilakukan dengan persamaan (6).

Misalkan: w*f* = [0.2, 0.3, 0.5], b*f* = b_f = [0.1, 0.1, 0.1], maka

$$ft = \sigma([0.2,0.3,0.5] \times [0,0,0,0.5,0.4,0.7] + [0.1,0.1,0.1])$$

$$ft = \sigma([0 + 0.2(0.5) + 0.3(0.4) + 0.5(0.7)] + [0.1])$$

$$ft = \sigma([0.1 + 0.06 + 0.35] + 0.1) = \sigma(0.61) = 0.647$$

b. Input Gate

Perhitungan *input gate* dilakukan dengan persamaan (7).

Misalkan W*i* = [0.5, 0.4, 0.3], b*i* = [0.2, 0.2, 0.2], maka

$$it = \sigma([0.5,0.4,0.3] \times [0,0,0,0.5,0.4,0.7] + [0.2])$$

$$it = \sigma([0.5(0.5) + 0.4(0.4) + 0.3(0.7)] + 0.2) = \sigma(0.25 + 0.16 + 0.21 + 0.2)$$
$$= \sigma(0.82) = 0.694$$

c. Update Cell State

Perhitungan *cell state* baru dilakukan dengan persamaan (8)

Dengan kandidat *cell state*: $Ct = tanh(Wc \times [ht - 1, xt] + bc)$

Misalkan W*c* = [0.3, 0.4, 0.6], b*c* = [0.1, 0.1, 0.1], maka

$$Ct = tanh([0.3,0.4,0.6] \times [0,0,0,0.5,0.4,0.7] + 0.1)$$

$$Ct = tanh(0.15 + 0.16 + 0.42 + 0.1) = tanh(0.83) = 0.68$$

Update *cell state*: $ct = 0.647 \times 0 + 0.694 \times 0.68 = 0.472$

***d. Output Gate* dan *Hidden State* Baru**

Perhitungan *output gate* dilakukan dengan persamaan (9).

Misalkan W*o* = [0.3, 0.6, 0.4], b*o* = [0.1, 0.1, 0.1]

$$ot = \sigma(0.3(0.5) + 0.6(0.4) + 0.4(0.7) + 0.1) = \sigma(0.63) = 0.652$$

Hidden state baru:

$$ht = ot \times tanh(ct) = 0.652 \times tanh(0.472) = 0.652 \times 0.439 = 0.286$$

Jadi sudah didapatkan:

- *Hidden state* pertama: 0.286
- *Cell state* pertama: 0.472

Proses ini diulang sampai timestep terakhir lalu output akhir diproses ke *Dense Layer* buat prediksi kategori. Hasil akhirnya:

- *Hidden state* terakhir: 0.724
- *Cell state* terakhir: 1.12

e. Dense Layer

Rumus:

$$Output = softmax(W \times ht + b) \quad (27)$$

Misalkan W = [0.5, 0.3, 0.2, 0.7, 0.4], b = [0.1, 0.1, 0.1, 0.1, 0.1]

$$Output = softmax([0.5,0.3,0.2,0.7,0.4] \times 0.724 + [0.1])$$

$$Output = softmax([0.362,0.217,0.144,0.507,0.289] + [0.1])$$

$$Output = softmax([0.462,0.317,0.244,0.607,0.389])$$

Softmax (ubah jadi probabilitas kategori):

$$softmax(xi) = \frac{exi}{\sum exj} \quad (28)$$

Hitung e untuk tiap nilai:

- $e^{0.462} = 1.587$
- $e^{0.317} = 1.373$
- $e^{0.244} = 1.276$
- $e^{0.607} = 1.835$
- $e^{0.389} = 1.475$

Total semua: 1.587+1.373+1.276+1.835+1.475 = 7.546

Akhirnya, didapatkan probabilitas kategori:

Tabel 3. 13 Probabilitas Kategori LSTM

Kategori	Probabilitas Akhir
0	$\frac{1.587}{7.546} = 0.210$
100	$\frac{1.373}{7.546} = 0.182$
200	$\frac{1.276}{7.546} = 0.169$

Tabel 3.13 (lanjutan)

300	$\frac{1.835}{7.546} = 0.243$
400	$\frac{1.475}{7.546} = 0.195$

Jadi kategori dengan probabilitas tertinggi untuk judul buku "*Analisis filsafat politik dalam pendidikan agama islam*" adalah Kategori 300 (Ilmu Sosial) dengan 0.243 atau 24.3%

3.3.4.2.7 Evaluasi Model LSTM

Setelah proses pelatihan selesai, tahap selanjutnya adalah melakukan evaluasi menyeluruh untuk mengukur kinerja model LSTM dan membandingkan beberapa model LSTM. Evaluasi ini bertujuan untuk memastikan model mencapai performa terbaiknya dalam melatih data dalam penelitian ini.

Proses evaluasi direncanakan mencakup dua metode analisis utama:

1. Analisis Kurva Pembelajaran (*Learning Curves*): Penelitian ini akan menganalisis grafik pergerakan nilai *Accuracy* dan *Loss* pada data latih (*training*) dan data validasi (*validation*) di setiap *epoch*. Analisis visual ini bertujuan untuk mendeteksi dini masalah stabilitas model, seperti indikasi *overfitting* (akurisasi *training* tinggi namun validasi rendah) atau *underfitting* (model gagal mempelajari pola), serta memastikan model mencapai titik konvergensi yang optimal
2. Pengukuran Metrik Kuantitatif: Model akhir akan diuji menggunakan Data Uji (*Testing Set*) untuk menghasilkan prediksi final. Hasil prediksi akan dipetakan dalam bentuk *Confusion Matrix* untuk menghitung metrik evaluasi standar klasifikasi, yaitu Akurasi, Presisi, *Recall*, dan *F1-Score*.

3.3.5 *Evaluation*

Tahap evaluasi dilakukan untuk mengukur performa dan membadingkan model Naive Bayes dan LSTM yang telah dikembangkan. Model diuji berdasarkan beberapa metrik evaluasi, yaitu akurasi *dan F1-score*. Akurasi mengukur seberapa

banyak prediksi yang benar dibandingkan total prediksi, cocok untuk dataset yang jumlahnya seimbang. *F1-Score* adalah rata-rata harmonik dari presisi dan *recall*, ini cocok saat dataset tidak seimbang.

Output dari tahap ini adalah model dengan performa terbaik yang telah dioptimasi berdasarkan hasil evaluasi, sehingga dapat digunakan untuk klasifikasi kategori buku dengan akurasi yang optimal.

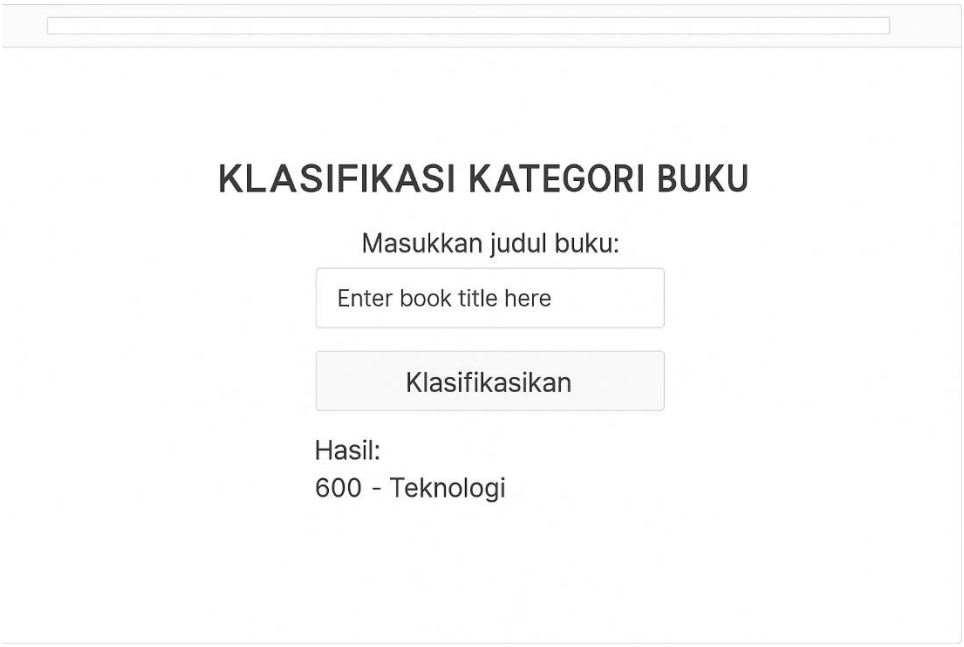
3.3.6 *Deployment*

Tahap ini bertujuan untuk mengimplementasikan model yang telah dikembangkan ke dalam sistem sederhana untuk menguji apakah model dapat berjalan dengan baik dalam skenario nyata. Model yang telah dievaluasi dan memiliki performa terbaik akan diterapkan dalam sebuah aplikasi berbasis web sederhana, di mana pengguna dapat memasukkan judul buku, lalu sistem akan menampilkan kategori dari buku tersebut.

Implementasi sistem ini dilakukan dengan menggunakan Flask untuk membangun *API* dan sebagai *web framework* yang berfungsi sebagai backend untuk menerima input dari pengguna dan menampilkan hasil prediksi kategori buku. Model klasifikasi yang telah dibuat sebelumnya menggunakan *Naive Bayes* dan LSTM akan disimpan dalam format *.pkl* (untuk *Naive Bayes*) dan *.h5* (untuk LSTM). Saat pengguna memasukkan judul buku, sistem akan melakukan prapemrosesan data seperti tokenisasi, penghapusan stopwords, dan representasi teks menggunakan TF-IDF atau *word embeddings*, sebelum akhirnya model melakukan prediksi dan menampilkan hasilnya di halaman web. Untuk tampilan halaman website akan dirancang menggunakan HTML, CSS, dan Javascript.

Tahap *deployment* dalam penelitian ini dibatasi hingga pengembangan antarmuka prototipe (*development*) yang berfungsi sebagai sarana uji coba fungsionalitas model. Aplikasi yang dibangun tidak ditujukan untuk diimplementasikan secara permanen pada sistem operasional UPA Perpustakaan Universitas Lampung, melainkan digunakan untuk memvalidasi apakah model yang telah dilatih dapat berfungsi dengan baik saat menerima input data baru secara langsung. Hal ini bertujuan untuk memastikan bahwa hasil penelitian memiliki nilai aplikatif dan

dapat memberikan gambaran mengenai potensi penggunaan model di masa mendatang.



Gambar 3. 2 *Wireframe* Antarmuka *Web Deployment* Sistem Klasifikasi Buku

Gambar 3.2 menampilkan rancangan awal (*wireframe*) antarmuka web sederhana yang digunakan dalam tahap *deployment* sistem. Antarmuka ini dirancang untuk memudahkan pengguna dalam melakukan uji coba model klasifikasi buku secara langsung. Pada tampilan utama, pengguna dapat memasukkan judul buku ke dalam form input yang telah disediakan. Setelah itu, sistem akan memproses input melalui model klasifikasi yang telah dilatih sebelumnya (*Naïve Bayes* dan LSTM) untuk memprediksi kategori buku. Hasil prediksi kemudian ditampilkan secara langsung pada halaman web.

Wireframe ini memberikan gambaran mengenai bagaimana sistem diimplementasikan dalam bentuk aplikasi berbasis web. Dengan adanya desain antarmuka ini, pengguna dapat lebih mudah memahami cara kerja sistem.

V. PENUTUP

5.1 Kesimpulan

Berdasarkan dari hasil penelitian, terdapat beberapa kesimpulan, yaitu:

1. Pemodelan *Naive Bayes* berhasil dikembangkan dengan pendekatan DFWNB menunjukkan hasil akurasi dan F1-Score 69% dengan waktu pelatihan hanya 1 detik, secara keseluruhan lebih baik dibandingkan varian *Naive Bayes* lainnya (MNB Standar, MNB + CRaic*,* LNB*, dan* GDNB). Namun, performa DFWNB masih berada di bawah model LSTM.
2. Pemodelan LSTM berhasil dikembangakan dengan hasil terbaik menggunakan arsitektur Bi-LSTM 1 layer dengan *word embedding* FastText 300 dimensi. Memberikan hasil dengan akurasi sebesar 79,3% dan rata-rata nilai F1-Score mendekati 0,73. Hal ini membuktikan bahwa pendekatan *Deep Learning* (Bi-LSTM) mampu memberikan performa klasifikasi yang lebih baik dibandingkan metode *Machine Learning* konvensional (DFWNB) dalam studi kasus klasifikasi judul buku ini.
3. Penelitian ini berhasil membangun sistem klasifikasi otomatis sederhana dengan menggunakan model Bi-LSTM yang dapat mengelompokkan judul buku ke dalam kategori secara cukup akurat dengan ketepatan prediksi 76,7% dari 30 buku yang diuji.

5.2 Saran

Saran untuk penelitian selanjutnya, yaitu:

1. Perluasan dataset dan atribut fitur. Pengembangan di masa depan sebaiknya menggunakan volume data yang lebih besar dan bervariasi dari berbagai repositori perpustakaan digital. Selain itu, integrasi atribut teks lain seperti

abstrak, sinopsis, atau daftar isi sangat direkomendasikan untuk memperkaya konteks semantik model dalam memahami isi buku.

2. Peningkatan kedalaman klasifikasi. Model klasifikasi dapat dikembangkan lebih lanjut hingga mencapai tingkat subkategori (*subclass*) dari sepuluh kelas utama DDC. Hal ini bertujuan agar sistem mampu memberikan pengelompokan yang lebih spesifik dan detail, sehingga membantu pustakawan dalam katalogisasi cabang ilmu yang lebih presisi
3. Optimisasi model lanjutan. disarankan untuk menerapkan teknik augmentasi data (*data augmentation*) pada dataset judul buku, khususnya pada kategori yang memiliki jumlah sampel terbatas (kelas minoritas) serta eksplorasi mendalam terhadap teknik *tuning hyperparameter* untuk mendapatkan konfigurasi model yang paling optimal dan efisien.

DAFTAR PUSTAKA

[1] F. Miao, P. Zhang, L. Jin, and H. Wu, "Chinese News Text Classification Based on Machine Learning Algorithm," in *2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, Hangzhou: IEEE, Aug. 2018, pp. 48–51. doi: 10.1109/IHMSC.2018.10117.

[2] X. Yu Li, L. B. Han, and Z. Feng Jiang, "Deep Learning-Based Algorithm for Classification of News Text," *IEEE Access*, vol. 12, pp. 159086–159098, 2024, doi: 10.1109/ACCESS.2024.3487311.

[3] Z. Li *et al.*, "A Unified Understanding of Deep NLP Models for Text Classification," *IEEE Trans. Vis. Comput. Graph.*, vol. 28, no. 12, pp. 4980–4994, Dec. 2022, doi: 10.1109/TVCG.2022.3184186.

[4] V. Ayumi, H. Noprisson, M. Utami, E. D. Putra, and M. Purba, *Konsep Dasar Natural Language Processing (NLP)*, 1st ed. Sukabumi: CV. Jejak, 2023. [Online]. Available: https://www.google.co.id/books/edition/Konsep_Dasar_Natural_Language_Processing/FTnhEAAAQBAJ?hl=id&gbpv=1&dq=Konsep+Dasar+Natural+Language+Processing+(NLP)&pg=PA125&printsec=frontcover

[5] M. Shamsitdinova, D. Khashimova, N. Niyazova, U. Nasirova, and N. Khikmatov, "Harnessing AI for Enhanced Searching in Digital Libraries: Transforming Research Practices," *Indian J. Inf. Sources Serv.*, vol. 14, no. 3, pp. 102–109, Sep. 2024, doi: 10.51983/ijiss-2024.14.3.14.

[6] C. R. A. Widiawati and A. M. Wahid, *Pemrosesan Bahasa Alami Konsep, Algoritma & Implementasi*, 1st ed. Banyumas, Jawa Tengah: Zahira Media Publisher, 2025. Accessed: Mar. 11, 2025. [Online]. Available: https://www.google.co.id/books/edition/PEMROSESAN_BAHASA_ALAMI_Konsep_Algoritma/bKlOEQAAQBAJ?hl=id&gbpv=1

[7] P. B. Wintoro, Khairudin, R. A. Pradipta, and Z. Huda, "Systematic Literature Review of Low Resource Text Classification," in *2025 International Conference on Converging Technology in Electrical and Information Engineering (ICCTEIE)*, 2025, pp. 212–217. doi: 10.1109/ICCTEIE66144.2025.11341775.

[8] Y. Borole, "Study on Feature Selection in Data Mining," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 7, no. 5, pp. 3956–3958, May 2019, doi: 10.22214/ijraset.2019.5652.

[9] R. T. Ginting, *Klasifikasi Sederhana dan Kompleks*. Tasikmalaya, Jawa Barat: Langgam Pustaka, 2025. [Online]. Available: https://www.google.co.id/books/edition/Klasifikasi_Sederhana_dan_Kompleks/oC6LEQAAQBAJ?hl=id&gbpv=0

[10] R. M. Sari, V. Tasril, S. Wahyuni, and S. E. Putri, *Klasifikasi Forecasting Menggunakan Algoritma Naive Bayes*, 1st ed. Payakumbuh: Serasi Media Teknologi, 2024. [Online]. Available: https://www.google.co.id/books/edition/Klasifikasi_Forecasting_Mengunakan_Algor/56oxEQAAQBAJ?hl=id&gbpv=1&dq=algoritma+naive+bayes+adalah&pg=PA19&printsec=frontcover

[11] B. P. Oyelakin, A. Adebiyi, B. Gbadamosi, A. M. Olaolu, A. Jesutofunmi, and A. Abidemi, "Text Classification Using Recurrent Neural Network and Support Vector Machine on a Customer Review Dataset," *J. Theor. Appl. Inf. Technol.*, vol. 100, no. 4, pp. 948–957, 2022.

[12] M. J. Alrawashdeh, "Improved Naive Bayes Classification for Joint Investment Plan," *WSEAS Trans. Math.*, vol. 21, pp. 37–43, Feb. 2022, doi: 10.37394/23206.2022.21.6.

[13] L. Afuan and R. Isnanto, *Deteksi Jatuh*, 1st ed. Banyumas, Jawa Tengah: Zahira Media Publisher, 2024. [Online]. Available: https://www.google.co.id/books/edition/Deteksi_jatuh/NKZOEQAAQBAJ?hl=id&gbpv=1&dq=daya+komputasi+naive+bayes+rendah&pg=PA54&printsec=frontcover

[14] P. J. B. Pajila, B. G. Sheena, A. Gayathri, J. Aswini, M. Nalini, and S. S. R, "A Comprehensive Survey on Naive Bayes Algorithm: Advantages, Limitations and Applications," in *2023 4th International Conference on Smart Electronics and Communication (ICOSEC)*, Trichy, India: IEEE, Sep. 2023, pp. 1228–1234. doi: 10.1109/ICOSEC58147.2023.10276274.

[15] Y. Asri, D. Kuswardani, L. F. M. Horhoruw, and S. A. Ramadhana, *Machine Learning & Deep Learning: Analisis Sentimen Menggunakan Ulasan Pengguna Aplikasi*, 1st ed. Ponorogo: Uwais Inspirasi Indonesia, 2024.

[16] S. Ruan, H. Li, C. Li, and K. Song, "Class-Specific Deep Feature Weighting for Naïve Bayes Text Classifiers," *IEEE Access*, vol. 8, pp. 20151–20159, 2020, doi: 10.1109/ACCESS.2020.2968984.

[17] T. Kim and J.-S. Lee, "Exponential Loss Minimization for Learning Weighted Naive Bayes Classifiers," *IEEE Access*, vol. 10, pp. 22724–22736, 2022, doi: 10.1109/ACCESS.2022.3155231.

[18] L. Jiang, C. Li, S. Wang, and L. Zhang, "Deep feature weighting for naive Bayes and its application to text classification," *Eng. Appl. Artif. Intell.*, vol. 52, pp. 26–39, Jun. 2016, doi: 10.1016/j.engappai.2016.02.002.

[19] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[20] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. in Adaptive Computation and Machine Learning series. MIT Press, 2016. [Online]. Available: https://books.google.co.id/books?id=Np9SDQAAQBAJ&hl=id&source=gbs_navlinks_s

[21] I. K. Ihianle, A. O. Nwajana, S. H. Ebenuwa, R. I. Otuka, K. Owa, and M. O. Orisatoki, “A Deep Learning Approach for Human Activities Recognition From Multimodal Sensing Devices,” *IEEE Access*, vol. 8, pp. 179028–179038, 2020, doi: 10.1109/ACCESS.2020.3027979.

[22] C. Liu, “Long short-term memory (LSTM)-based news classification model,” *PLOS ONE*, vol. 19, no. 5, p. e0301835, May 2024, doi: 10.1371/journal.pone.0301835.

[23] U. M. Indonesia *et al.*, “MODEL BIDIRECTIONAL LSTM UNTUK PEMROSESAN SEKUENSIAL DATA TEKS SPAM,” *METHOMIKA J. Manaj. Inform. Dan Komputerisasi Akunt.*, vol. 7, no. 2, pp. 265–271, Oct. 2023, doi: 10.46880/jmika.Vol7No2.pp265-271.

[24] F. Sulianta, *Dasar & Konsep Deep Learning*, 1st ed. Online: Feri Sulianta, 2024. [Online]. Available: https://www.google.co.id/books/edition/Dasar_Konsep_Deep_Learning/FpI2EQAAQBAJ?hl=id&gbpv=0&kptab=overview

[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[26] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” Mar. 02, 2015, *arXiv*: arXiv:1502.03167. doi: 10.48550/arXiv.1502.03167.

[27] M. Sanjaya W. S., Ph.D., *Deep Learning Convolutional Neural Networks: Pemrograman Python + Arduino serta Aplikasi pada Komputer Vision, Speech Recognition, dan Mobile Robot - Penerbit Bolabot*, 1st ed. Bandung: Bolabot, 2024.

[28] J. Ahmed Bhutto, M. Aamir, U. Aslam Bhatti, W. Ahmed Abro, and Z. Rahman, *Deep Learning in Medical Signal and Image Processing*. IGI Global, 2025.

[29] M. Sanjaya and D. Anggraeni, *Penerapan Deep Learning untuk Robot Arm Sortasi Warna berbasis Pemrograman Python + IDE Arduino - Penerbit Bolabot*, 1st ed. Bandung: Bolabot, 2025. [Online]. Available:

https://www.google.co.id/books/edition/Penerapan_Deep_Learning_untuk_Robot_Arm/bx2OEQAAQBAJ?hl=id&gbpv=1

[30] D. O. Sihombing, "Implementasi Natural Language Processing (NLP) dan Algoritma Cosine Similarity dalam Penilaian Ujian Esai Otomatis," *J. Sist. Komput. Dan Inform. JSON*, vol. 4, no. 2, p. 396, Dec. 2022, doi: 10.30865/json.v4i2.5374.

[31] J. W. Tukey, *Exploratory Data Analysis*. Addison-Wesley, 1977.

[32] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques (3rd ed.)*, 3rd ed. Morgan Kaufmann. [Online]. Available: https://drive.google.com/file/d/0BwxUBHTpU9kCZ0JERnJaWDBDN2c/view?resourcekey=0-71i7Q8dA8qYekqpeefoASQ

[33] C. Manning, P. Raghavan, and H. Schuetze, "Introduction to Information Retrieval," *Camb. Univ. Press*, 2008.

[34] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 40, no. 3, pp. 211–218, Jul. 2006, doi: 10.1108/00330330610681286.

[35] M. Adriani, J. Asian, B. A. A. Nazief, S. M. M. Tahaghoghi, and H. E. Williams, "Stemming Indonesian: A confix-stripping approach," *ACM Trans. Asian Lang. Inf. Process. TALIP*, vol. 6, no. 4, pp. 1–33, 2007, doi: https://doi.org/10.1145/1316457.1316459.

[36] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inf. Process. Manag.*, vol. 24, no. 5, pp. 513–523, Jan. 1988, doi: 10.1016/0306-4573(88)90021-0.

[37] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," Jun. 05, 2018, *arXiv*: arXiv:1201.0490. doi: 10.48550/arXiv.1201.0490.

[38] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," 2013, *arXiv*. doi: 10.48550/ARXIV.1301.3781.

[39] J. Pennington, R. Socher, and C. Manning, "Glove: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, 2014, pp. 1532–1543. doi: 10.3115/v1/D14-1162.

[40] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov, "Learning Word Vectors for 157 Languages," 2018, doi: https://doi.org/10.48550/arXiv.1802.06893.

[41] P. W. Rahayu *et al.*, *Buku Ajar Data Mining*, 1st ed. Jambi: PT. Sonpedia Publishing Indonesia, 2024. [Online]. Available:

https://www.google.co.id/books/edition/Buku_Ajar_Data_Mining/vCruEAAAQBAJ?hl=id&gbpv=1

[42] L. Hakim, A. Sobri, L. Sunardi, and D. Nurdiansyah, "Prediksi penyakit jantung berbasis mesin learning dengan menggunakan metode k-nn," *J. Digit. Teknol. Inf.*, vol. 7, no. 2, p. 14, Feb. 2025, doi: 10.32502/digital.v7i2.9429.

[43] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, C. Shearer, and R. Wirth, *CRISP-DM 1.0: Step-by-Step Data Mining Guide*. SPSS Inc., 2000.

[44] G. T. Wicaksono, *Explore Informatika untuk SMP/MTs Kelas VII*. Bandung: Penerbit Duta, 2019. [Online]. Available: https://www.google.co.id/books/edition/Explore_Informatika_untuk_SMP_MTs_Kelas/3rhHEAAAQBAJ?hl=id&gbpv=1

[45] S. Maesaroh, Afiyati, L. Hakim, Y. S. Sari, and M. Yusuf, *Bahasa Pemrograman Python*. Banten: Sada Kurnia Pustaka, 2024. [Online]. Available: https://www.google.co.id/books/edition/Bahasa_Pemrograman_Python/bOIKEQAAQBAJ?hl=id&gbpv=1

[46] B. Zuhri and N. H. Harani, *Aplikasi Rekrutmen Karyawan Dengan Algoritma Artificial Neural Network (ANN) Dan Framework Flask*, 1st ed. Bandung: Penerbit Buku Pedia, 2023. [Online]. Available: https://www.google.co.id/books/edition/Aplikasi_Rekrutmen_Karyawan_Dengan_Algor/yeCuEAAAQBAJ?hl=id&gbpv=1

[47] W. B. Zulfikar, A. R. Atmadja, and S. F. Pratama, "Sentiment Analysis on Social Media Against Public Policy Using Multinomial Naive Bayes," *Sci. J. Inform.*, vol. 10, no. 1, pp. 25–34, Jan. 2023, doi: 10.15294/sji.v10i1.39952.

[48] Y. Qixuan, "Three-Class Text Sentiment Analysis Based on LSTM," 2024, *arXiv*. doi: 10.48550/ARXIV.2412.17347.

[49] G. Singh, B. Kumar, L. Gaur, and A. Tyagi, "Comparison between Multinomial and Bernoulli Naïve Bayes for Text Classification," in *2019 International Conference on Automation, Computational and Technology Management (ICACTM)*, London, United Kingdom: IEEE, Apr. 2019, pp. 593–596. doi: 10.1109/ICACTM.2019.8776800.

[50] G. Dou, K. Zhao, M. Guo, and J. Mou, "Memristor-Based LSTM Network for Text Classification," *Fractals*, vol. 31, no. 06, p. 2340040, Jan. 2023, doi: 10.1142/S0218348X23400406.

[51] M. Abbas, A. Kamran, Memon, A. A. Jamali, Saleemullah Memon, and Anees Ahmed, "Multinomial Naive Bayes Classification Model for Sentiment Analysis," 2019, *International Journal of Computer Science and Network Security*. doi: 10.13140/RG.2.2.30021.40169.

[52] N. Benarkah, V. R. Prasetyo, and J. R. Soetiyono, “Sentiment Analysis for Sumber Gempong Rice Field-Based Tourism Destination using Long Short-Term Memory,” *Keluwih J. Sains Dan Teknol.*, vol. 5, no. 2, Oct. 2024, doi: 10.24123/saintek.v5i2.6498.

[53] A. H. Odeh, M. Odeh, H. Odeh, and N. Odeh, “Using Natural Language Processing for Programming Language Code Classification with Multinomial Naive Bayes,” *Rev. Intell. Artif.*, vol. 37, no. 5, pp. 1229–1236, Oct. 2023, doi: 10.18280/ria.370515.

[54] A. C. Nurcahyo, A. T. H. Yong, and A. F. Atanda, “Classification of Simulated Fake Bandwidth Data Using LSTM,” *TEPIAN*, vol. 5, no. 3, pp. 35–47, Sep. 2024, doi: 10.51967/tepian.v5i3.3106.

[55] C. Dewi, R.-C. Chen, H. J. Christanto, and F. Cauteruccio, “Multinomial Naïve Bayes Classifier for Sentiment Analysis of Internet Movie Database,” *Vietnam J. Comput. Sci.*, vol. 10, no. 04, pp. 485–498, Nov. 2023, doi: 10.1142/S2196888823500100.

[56] J. R. Landis and G. G. Koch, “The Measurement of Observer Agreement for Categorical Data,” *Biometrics*, vol. 33, no. 1, p. 159, Mar. 1977, doi: 10.2307/2529310.