PENERAPAN *BACKPROPAGATION NEURAL NETWORK* DAN SMOTE PADA ANALISIS SENTIMEN ULASAN PENGGUNA *MYXL* DI *GOOGLE PLAY STORE*

THESIS

Oleh:
BADRIYAH
NIM. 200605220010

PROGRAM STUDI MAGISTER INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2024

PENERAPAN *BACKPROPAGATION NEURAL NETWORK* DAN SMOTE PADA ANALISIS SENTIMEN ULASAN PENGGUNA *MYXL* DI *GOOGLE PLAY STORE*

THESIS

Diajukan kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Magister Komputer (M. Kom.)

Oleh:
BADRIYAH
NIM. 200605220010

PROGRAM STUDI MAGISTER INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2024

PENERAPAN *BACKPROPAGATION NEURAL NETWORK* DAN SMOTE PADA ANALISIS SENTIMEN ULASAN PENGGUNA *MYXL* DI *GOOGLE PLAY STORE*

THESIS

Diajukan kepada:
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Magister Komputer (M. Kom.)

Oleh:
BADRIYAH
NIM. 200605220010

PROGRAM STUDI MAGISTER INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2024

PENERAPAN *BACKPROPAGATION NEURAL NETWORK* DAN SMOTE PADA ANALISIS SENTIMEN ULASAN PENGGUNA *MYXL* DI *GOOGLE PLAY STORE*

THESIS

Oleh:
BADRIYAH
NIM. 200605220010

Telah diperiksa dan disetujui untuk diuji:
Tanggal: 17 Oktober 2024

Pembimbing I,	Pembimbing II,
Dr. Hbtok Chamidy, M. Kom.	Prof. Dr. Suhartono, M. Kom.
NIP. 19691222 200604 1 001	NIP. 19680519 200312 1 001

Mengetahui,
Ketua Program Studi Magister Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

PENERAPAN *BACKPROPAGATION NEURAL NETWORK* DAN SMOTE PADA ANALISIS SENTIMEN ULASAN PENGGUNA *MYXL* DI *GOOGLE PLAY STORE*

THESIS

Oleh:
BADRIYAH
NIM. 200605220010

Telah Dipertahankan di Depan Dewan Penguji Thesis
Dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Magister Komputer (M. Kom.)
Tanggal: 17 Oktober 2024

Susunan Dewan Penguji			**Tanda Tangan**
Penguji I	:	Dr. M. Ainul Yaqin, M. Kom NIP. 19761013 200604 1 004	( )
Penguji II	:	Dr. Imamuddin, Lc., MA. NIP. 19740602 200901 1 010	( )
Pembimbing I	:	Dr. Totok Chamidy, M. Kom NIP. 19691222 200604 1 001	( )
Pembimbing II	:	Prof. Dr. Suhartono, M. Kom. NIP. 19680519 200312 1 001	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Magister Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini:

Nama	: Badriyah
NIM	: 200605220010
Program Studi	: Magister Informatika
Fakultas	: Sains dan Teknologi

Menyatakan dengan sebenarnya bahwa Thesis yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambilalihan data, tulisan atau pikiran orang lain yang saya akui sebagai tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila di kemudian hari terbukti atau dapat dibuktikan Thesis ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 17 Oktober 2024
Yang membuat pernyataan

Badriyah
NIM. 200605220010

MOTTO

"Selalu ikhlas dan bersyukur atas rahman dan rahim-Nya"

PERSEMBAHAN

Penulis persembahkan Thesis ini untuk suami, mertua, ibu dan anak-anakku terkasih. Terima kasih tak terhingga atas doa dan dukungan yang telah diberikan hingga selesainya studi ini.

KATA PENGANTAR

Assalaamu ‘alaikum Wr. Wb.

Alhamdulillahi Robbil ‘Aalamiin, yang telah melimpahkan rahmat, taufiq dan hidayah-Nya, sehingga penulis dapat menyelesaikan Thesis sebagai syarat kelulusan di Program Studi Magister Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang ini dengan baik.

Ucapan terima kasih dengan do’a *jazakummullahu khoiron katsiron*, penulis sampaikan kepada:

1. Bapak Dr. Totok Chamidy, M. Kom. dan Bapak Prof. Dr. Suhartono, M.Kom. selaku dosen pembimbing Thesis, yang telah banyak memberikan pengarahan dan pengalaman yang berharga.
2. Segenap civitas akademika Program Studi Magister Informatika, terutama seluruh Bapak/Ibu dosen yang telah banyak memberikan ilmu dan bimbingan.
3. Keluarga tercinta, yang senantiasa memberikan do’a dan.dukungan.
4. Rekan-rekan seangkatan yang selalu berbagi ilmu dan pengalaman.

Penulis menyadari bahwa Thesis ini masih jauh dari kata sempurna, namun penulis berharap semoga Thesis ini dapat memberi manfaat kepada pembaca.

Wassalaamu’alaikum Wr. Wb.

Malang, 17 Oktober 2024

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Badriyah, 2024. **Penerapan *Backpropagation Neural Network* Dan SMOTE Pada Analisis Sentimen Ulasan Pengguna *MyXL* di *Google Play Store*.** Thesis. Program Studi Magister Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Dr. Totok Chamidy, M.Kom. (II) Prof. Dr. Suhartono, M.Kom.

Kata kunci: analisis sentimen, *Backpropagation Neural Network*, SMOTE.

Opini pelanggan sangat penting bagi suatu perusahaan untuk memahami produk atau layanannya. Analisis sentimen membantu perusahaan mendapatkan wawasan lebih dalam dari ulasan pelanggan. Namun, ketidakseimbangan data seringkali menjadi tantangan. Model tersebut cenderung mengakui kelas mayoritas sehingga mengabaikan kelas minoritas dan membuat hasil analisis menjadi bias. Penelitian ini mengkaji sentimen ulasan pengguna aplikasi *MyXL* di *Google Play Store* dengan menggunakan teknik SMOTE untuk menangani ketidakseimbangan data, dan algoritma *Backpropagation Neural Network* sebagai model klasifikasinya. Optimasi model dilakukan melalui pelatihan awal menggunakan teknik validasi silang dan penyetelan *hyperparameter* manual. Pelatihan selanjutnya dilakukan dengan *fine-tuning* nilai di sekitar *learning rate* yang menghasilkan performa model terbaik pada pelatihan awal. Fine-tuning menghasilkan performa model paling optimal dengan *accuracy* 74% dan F1-*score* 69,71% pada nilai *learning rate* antara 0,0098 hingga 0,0104 menggunakan kombinasi parameter dua *hidden layer* dan *neuron* (128,64).

ABSTRACT

Badriyah, 2024. **Implement Backpropagation of Neural Network and SMOTE on Sentiment Analysis of MyXL User Reviews on Google Play Store**. Thesis. Master of Informatics Study Program, Faculty of Science and Technology, State Islamic University of Maulana Malik Ibrahim Malang. Advisors: (I) Dr. Totok Chamidy, M.Kom. (II) Prof. Dr. Suhartono, M.Kom.

Keywords: sentiment analysis, Backpropagation Neural Network, SMOTE.

Customer opinion is very important for a company to understand its products or services. Sentiment analysis helps companies gain deeper insights from customer reviews. However, data imbalance is often a challenge. The model tends to recognize the majority class, thereby ignoring the minority class and making the analysis results biased. This research examines the sentiment of user reviews of the MyXL application on the Google Play Store using the SMOTE technique to handle data imbalance, and the Backpropagation Neural Network algorithm as a classification model. Model optimization was done through initial training using cross-validation techniques and manual hyperparameter tuning. Subsequent training is performed by fine-tuning the values around the learning rate which results in the best model performance on initial training. Fine-tuning produces the most optimal model performance with an accuracy of 74% and an F1 score of 69.71% at learning rate values between 0.0098 and 0.0104 using a combination of two hidden layer parameters and neurons (128,64).

الملخص

بدرية، 2024. تطبيق الشبكة العصبية للانتشار العكسي و تقنية أخذ العينات الزائدة للأقلية الاصطناعية (SMOTE) في تحليل مشاعر مراجعة مستخدم *MyXL* على متجر *Google Play*. رسالة الماجستير. قسم المعلومات، كلية العلوم والتكنولوجيا بجامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج. المشرف الأول: د. توتوك حميدي، الماجستير. المشرف الثاني: أ.د. سوهارتونو، الماجستير.

الكلمات الرئيسية: تحليل المشاعر، الشبكة العصبية للانتشار العكسي، تقنية أخذ العينات الزائدة للأقلية الاصطناعية.

آراء العملاء ضرورية للشركات لفهم منتجاتها أو خدماتها. يساعد تحليل المشاعر الشركات على اكتساب رؤى أعمق من مراجعات العملاء. ومع ذلك، غالبا ما تشكل اختلالات البيانات تحديا. من المرجح أن يعترف النموذج بطبقة الأغلبية، وبالتالي يتجاهل طبقة الأقلية ويجعل نتائج تحليل التحيز. بحثت هذه الرسالة في مشاعر مراجعات مستخدمي تطبيق *MyXL* على متجر *Google Play* باستخدام تقنية SMOTE للتعامل مع اختلالات البيانات، وخوارزمية الشبكة العصبية للانتشار العكسي كنموذج تصنيف. تم تحسين النموذج من خلال التدريب الأولي باستخدام تقنية التحقق المتقاطع والضبط اليدوي للمعلمات الفائقة. تم تنفيذ التدريب التالي عن طريق الضبط الدقيق للقيمة حول معدل التعلم الذي أنتج أفضل أداء نموذجي في التدريب الأولي. ينتج الضبط الدقيق أفضل أداء للنموذج بدقة 74٪ ودرجة ف1 تبلغ 69.71٪ بمعدل تعلم يتراوح بين 0.0098 إلى 0.0104 باستخدام مجموعة من معلمات طبقتين مخفيتين و خلية عصبية (128.64).

BAB I

PENDAHULUAN

A. Latar Belakang

Natural Language Processing (NLP) merupakan cabang dari kecerdasan buatan (AI) yang memungkinkan komputer dapat memahami, menganalisis dan memproduksi bahasa manusia dengan cara yang cerdas (IPP and M.Si, 2024). Dengan kemampuan ini, komputer bisa berkomunikasi secara alami menggunakan bahasa seperti yang dilakukan manusia. Meski begitu, pengembangan NLP bukanlah proses yang mudah. Dibutuhkan sumber daya yang besar serta investasi signifikan untuk mencapai tingkat kecerdasan komputer yang mendekati kemampuan manusia.

Pengembangan ilmu NLP didasarkan pada ide untuk memberikan kemampuan kepada komputer agar bisa memahami dan merespons teks atau bahasa seperti kecerdasan yang telah Allah SWT anugerahkan kepada manusia. Hal ini selaras dengan firman Allah SWT dalam Al-Qur'an, Surat Al-Qiyamah: 16-19.

لَا تُحَرِّكْ بِهٖ لِسَانَكَ لِتَعْجَلَ بِهٖۗ ۝ اِنَّ عَلَيْنَا جَمْعَهٗ وَقُرْاٰنَهٗۚ ۝ فَاِذَا قَرَأْنٰهُ فَاتَّبِعْ قُرْاٰنَهٗۚ ۝ ثُمَّ اِنَّ عَلَيْنَا بَيَانَهٗۗ ۝

Janganlah engkau (Nabi Muhammad) gerakkan lidahmu (untuk membaca Al-Qur'an) karena hendak tergesa-gesa (menguasai)-nya. Sesungguhnya tugas Kamilah mengumpulkannya (dalam hatimu) dan membacakannya. Maka, apabila Kami telah selesai membacakannya, ikutilah bacaannya itu. Kemudian, sesungguhnya tugas Kami (pula)-lah untuk menjelaskannya.

Nabi Muhammad SAW merasakan rasa pengagungan yang mendalam terhadap Al-Qur'an sebagai tuntunan yang sangat mulia, serta sebagai kalam Tuhan Yang Maha Agung. Hal ini menyebabkan Beliau mengalami kesulitan di awal

penerimaan wahyu (Shihab, 2021). Ketika malaikat Jibril AS membacakan wahyu, Nabi Muhammad bersegera mengikuti dengan menggerakkan lidahnya, dengan tujuan untuk menghafal dan memastikan tidak ada bacaan yang terlewatkan. Nabi begitu cemas sehingga berusaha mengikuti bacaan Al-Qur'an, bahkan sebelum Jibril selesai membacanya, karena khawatir akan ada bagian yang terlupa.

Allah SWT kemudian menurunkan firman-Nya, menegaskan bahwa pengumpulan Al-Qur'an di dalam dada Rasul dan menjadikan Beliau hafal, fasih, serta lancar dalam membacanya adalah tanggung jawab Allah SWT. Nabi Muhammad SAW dilarang untuk mendahului malaikat Jibril ketika menyampaikan wahyu. Setelah Jibril selesai membacakan Al-Qur'an, Allah memerintahkan Nabi untuk mengikuti bacaan tersebut dengan penuh perhatian (melibatkan lidah, pendengaran, hati, dan pikiran) dan mengamalkannya dengan sungguh-sungguh.

Ayat-ayat dalam QS Al-Qiyamah:16-19 menunjukkan bahwa Allah telah mengaruniakan kemampuan luar biasa pada Nabi Muhammad SAW --dan pada manusia secara umum-- untuk menyerap, mengingat, memahami dan menafsirkan bahasa dengan baik. Karunia Allah berupa organ otak manusia telah dirancang untuk memiliki kemampuan itu semua. Dalam NLP, organ ini telah di-replikasi dalam bentuk algoritma yang dirancang untuk menganalisis teks dan memproses makna yang ada di dalamnya. Walaupun NLP belum mencapai kompleksitas otak manusia, namun penelitian NLP ini berusaha meniru sebagian dari cara kerja otak untuk memahami, menyimpan, dan menggunakan informasi.

Analisis sentimen adalah salah satu penerapan penting dalam pemrosesan bahasa alami (NLP) yang berfokus pada identifikasi informasi subjektif dari ulasan

berbasis teks (Amrullah et al., 2020). Proses ini memungkinkan kita untuk secara otomatis mengekstraksi, memahami, dan mengolah data teks tidak terstruktur dengan tujuan mengidentifikasi sentimen di balik sebuah pernyataan atau opini (Arsi and Waluyo, 2021). Cabang studi ini mencakup pemahaman terhadap opini, evaluasi, penilaian, sikap, dan emosi publik terhadap beragam entitas, seperti produk, layanan, organisasi, individu, isu, peristiwa, dan topik, serta atribut-atributnya (Asrumi et al., 2023). Analisis sentimen menjadi solusi efektif untuk mengelompokkan opini publik secara otomatis menjadi sentimen positif, negatif, atau netral (Rahayu et al., 2022). Dalam beberapa kasus, jika sebuah sentimen mengandung kata-kata yang saling bertentangan, maka sentimen tersebut bisa dikategorikan sebagai positif atau negatif. Namun, jika tidak ada keberpihakan yang jelas, sentimen tersebut akan dianggap netral (Noor and Turan, 2019). Dengan demikian, analisis sentimen adalah metode otomatis dalam NLP yang digunakan untuk memahami opini publik terhadap berbagai entitas --mulai dari produk hingga isu sosial-- dan mengelompokkan opini tersebut menjadi sentimen positif, negatif, atau netral berdasarkan analisis subjektif terhadap teks.

Opini pelanggan merupakan indikator penting dalam menilai keberhasilan suatu layanan atau produk. Dengan menggunakan analisis sentimen, perusahaan dapat memperoleh wawasan mendalam mengenai persepsi pelanggan terhadap produk atau layanan mereka. Di sisi lain, Hasibuan & Heriyanto (2022) menjelaskan bahwa opini pelanggan tidak hanya bermanfaat bagi perusahaan, tetapi juga berfungsi sebagai informasi berharga bagi pelanggan lainnya. Sebuah survei terhadap lebih dari 7.000 konsumen di 11 wilayah Asia Pasifik menunjukkan bahwa

76% konsumen mencari ulasan terlebih dahulu dan memverifikasi reputasi perusahaan sebelum melakukan pembelian (PricewaterhouseCoopers, n.d.).

Pengalaman pelanggan kini memiliki pengaruh signifikan terhadap keputusan transaksi seseorang. Pengalaman pelanggan dari ulasan pengguna produk atau jasa, baik dari instansi pemerintah maupun swasta, kelompok atau individu, sering ditemukan pada berbagai aplikasi di *Google Play Store*. *Play Store* sebagai *platform* resmi *Google* untuk aplikasi *Android*, menyediakan fitur ulasan di mana pengguna dapat menuliskan pengalaman mereka setelah menggunakan aplikasi. Menurut sumber ("*Google Play*," n.d.), ulasan pelanggan adalah cara yang efektif untuk berbagi pengalaman dan membantu pengguna lain dalam menentukan pilihan aplikasi yang sesuai dengan kebutuhan mereka.

B. Pernyataan Masalah

Permasalahan yang akan diselesaikan dalam penelitian ini adalah bagaimana cara mengukur tingkat kepuasan pelanggan aplikasi *MyXL* berdasarkan ulasan pengguna yang terdapat pada *Google Play Store* dengan menggunakan metode analisis sentimen.

C. Tujuan Penelitian

Penelitian ini bertujuan untuk:

1. Mengatasi tantangan dalam analisis sentimen ulasan pengguna *MyXL* di *Google Play Store*, terutama terkait dengan ketidakseimbangan data.
2. Membangun *Backpropagation Neural Network* dan teknik SMOTE dalam analisis sentimen ulasan pengguna *MyXL* di *Google Play Store* guna mendapatkan akurasi klasifikasi dengan cara yang optimal.

D. Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat kepada:

1. Peneliti, dengan memberikan pengalaman berharga dalam memahami dan melaksanakan proses pembuatan model *Backpropagation Neural Network* dengan menerapkan SMOTE dalam analisis sentimen ulasan pengguna *MyXL* di *Google Play Store*.
2. Pembaca, model *Backpropagation Neural Network* yang menerapkan SMOTE dalam analisis sentimen ulasan pengguna *MyXL* di *Google Play Store* ini diharapkan dapat menjadi referensi yang berguna bagi penelitian-penelitian selanjutnya.
3. PT XL Axiata Tbk, hasil penelitian ini dapat dijadikan rekomendasi untuk meningkatkan mutu produk dan layanan pelanggan, dengan memanfaatkan model *Backpropagation Neural Network* yang telah dikembangkan untuk menganalisis ulasan pengguna *MyXL* di *Google Play Store.*

E. Batasan Masalah

Agar penelitian ini lebih terarah, peneliti menggunakan beberapa batasan sebagai berikut:

1. Dataset yang digunakan dalam penelitian ini dapat diunduh dari https://kaggle.com/code/dimasdiandraa/analisis-sentimen-ulasan-myxl-dengan-svm
2. *Hyperparameter* yang digunakan pada penelitian ini adalah *learning rate* jumlah *hidden layer* dan jumlah *neuron*.

BAB II

STUDI PUSTAKA

A. Aplikasi *MyXL* di *Google Play Store*

PT. XL Axiata Tbk, atau XL Axiata, adalah perusahaan swasta yang bergerak di bidang layanan telekomunikasi di Indonesia. Perusahaan ini telah beroperasi sejak 8 Oktober 1996 ("Tentang XL Axiata," n.d.). Salah satu fasilitas yang disediakan perusahaan ini adalah aplikasi *MyXL*, yang tersedia di *Google Play Store* sebagai aplikasi resmi untuk pengguna XL. Aplikasi ini memungkinkan pengguna untuk memeriksa kuota dan pulsa secara gratis.

Google Play Store mencatat bahwa aplikasi *MyXL* telah diunduh lebih dari 50 juta kali. Sebagai *platform* resmi untuk sistem operasi *Android*, *Play Store* menyediakan fitur ulasan, di mana pengguna dapat menuliskan pengalaman mereka dengan aplikasi tersebut. Menurut sumber ("Google Play," n.d.), ulasan merupakan cara yang efektif untuk berbagi pengalaman dan membantu orang lain dalam menentukan pilihan aplikasi yang sesuai.

Penyediaan aplikasi *MyXL* di *Google Play Store* oleh PT. XL Axiata Tbk menjadi sarana penting dalam memberikan kemudahan akses bagi para pengguna jasa perusahaan, serta memperkuat interaksi antara perusahaan dan pelanggan melalui umpan balik yang diperoleh dari ulasan pengguna.

B. Analisis Sentimen

Menurut Purnamasari and Aji (2023), analisis sentimen adalah teknik pengolahan data teks yang secara otomatis mengekstrak ungkapan perasaan penulis, baik itu positif, negatif, atau netral, dengan menggunakan teknologi

Natural Language Processing (NLP) dan *machine learning*. Sebagian besar data yang digunakan dalam penelitian analisis sentimen berasal dari media sosial, di mana pengguna secara aktif mengekspresikan pendapat mereka, sehingga menghasilkan data yang melimpah.

Perkembangan penelitian tentang analisis sentimen sejalan dengan kebutuhan berbagai pihak akan pendapat publik mengenai topik tertentu. Analisis sentimen sangat berguna bagi individu dalam pengambilan keputusan, terutama saat memilih produk atau jasa, karena keputusan tersebut sering kali dipengaruhi oleh opini dan penilaian orang lain. Dari perspektif perusahaan, analisis sentimen dapat membantu meningkatkan kualitas produk dan layanan mereka.

Hidayat et al. (2021) menjelaskan bahwa analisis sentimen adalah bagian dari *text mining* yang berfokus pada studi dan klasifikasi opini, emosi, dan sentimen secara komputasi. Analisis ini sering disebut sebagai *opinion mining* dan berkaitan erat dengan topik *information retrieval*. Namun, perbedaan mendasar antara keduanya adalah bahwa algoritma *information retrieval* bekerja dengan data faktual, sementara penambangan opini berfokus pada data subjektif. Analisis sentimen juga termasuk dalam lingkup NLP, yang mempelajari aspek-aspek linguistik. Dalam analisis sentimen, terdapat proses komputasi yang mengklasifikasikan polaritas teks menjadi positif, negatif, atau netral, sehingga sering juga dikenal sebagai klasifikasi sentimen.

Yudianto et al. (2022) menyoroti bahwa klasifikasi sentimen dapat dilakukan dengan dua pendekatan utama, yaitu pendekatan berbasis leksikon yang menggunakan kamus untuk menentukan polaritas, dan pendekatan *machine*

learning yang memerlukan dataset berlabel dengan anotasi manual. Selain itu, terdapat juga pendekatan *hybrid* yang menggabungkan kedua metode tersebut.

Berdasarkan sumber data, Matondang et al. (2024) membedakan analisis sentimen menjadi tiga tingkatan: analisis sentimen tingkat dokumen, kalimat, dan aspek. Analisis sentimen tingkat dokumen mengklasifikasikan ulasan secara keseluruhan sebagai satu kesatuan yang memiliki polaritas positif atau negatif. Analisis sentimen tingkat kalimat fokus pada ekstraksi pendapat atau sentimen yang diungkapkan dalam kalimat tertentu. Sedangkan analisis sentimen tingkat aspek mengukur opini terhadap karakteristik spesifik dari suatu entitas.

C. *Text Analytics*, *Text Mining* dan *Data Mining*

Istilah *text analytics* seringkali disamakan dengan *text mining* atau *text data mining*, di mana *text mining* merujuk pada proses mengekstrak data dari teks. Sehingga, *text analytics*, *text mining*, dan *text data mining* sering digunakan secara bergantian. Menurut Annissa and Ariesta (2023) *text analytics* merupakan proses penemuan pola, tema, dan topik dari data yang tidak terstruktur.

Alex Yu (2022) mendefinisikan *text mining* sebagai proses transformasi teks yang tidak terstruktur menjadi format terstruktur, yang memungkinkan untuk mengekstraksi pola bermakna dan wawasan baru dari sejumlah data yang besar. Ia juga menjelaskan bahwa *data mining* adalah aplikasi spesifik dari ilmu data yang mencakup proses atau metode untuk mengekstrak pengetahuan atau pola menarik dari data besar. Sementara *data mining* lebih banyak berfokus pada data terstruktur dan numerik, *text mining* lebih diarahkan pada data yang tidak terstruktur atau semi-terstruktur. Meskipun terdapat perbedaan ini, kedua metode tersebut memiliki

tujuan yang sama, yaitu mengekstrak pola bermakna dari data besar dengan menggunakan teknik yang sebanding. Berbagai teknik dari *data mining*, seperti faktorisasi matriks, pengelompokan, dan klasifikasi, juga diterapkan dalam *text mining.*

Text analytics memiliki berbagai aplikasi yang luas, terkait dengan banyak disiplin ilmu seperti Statistik, Kecerdasan Buatan (AI), Pengolahan Bahasa Alami (NLP), *Data Mining* (DM), Pembelajaran Mesin (ML), dan Pengambilan Informasi (IR). Menurut Wikipedia ("*Text mining*," 2023), *text analytics* menjelaskan serangkaian teknik linguistik, statistik, dan pembelajaran mesin yang memodelkan serta menyusun konten informasi dari sumber tekstual untuk keperluan intelijen bisnis, analisis data eksploratif, penelitian, atau investigasi.

D. Natural Language Processing

Menurut Campesato (2021), *Natural Language Processing* (NLP) adalah cabang dari *Artificial Intelligence* (AI) yang mempelajari pemrosesan bahasa manusia oleh mesin (komputer). Aplikasi NLP seperti asisten suara, mesin pencari, terjemahan mesin, pemrosesan teks ringkasan, dan pengenalan suara, memiliki peranan penting dalam menyederhanakan berbagai tugas dan aspek kehidupan sehari-hari manusia.

Text mining memanfaatkan teknologi NLP, yang merupakan sub-bidang dari AI dan *Computational Linguistics* (CL). NLP adalah teknik yang bertujuan untuk memfasilitasi komunikasi antara manusia dan mesin (komputer). Menurut Otter et al. (2019), NLP juga dikenal sebagai *computational linguistics*, yang melibatkan rekayasa model komputasi dan proses untuk memecahkan masalah

dalam memahami bahasa manusia.

Galassi et al. (2021) menjelaskan bahwa NLP adalah aplikasi ilmu komputer yang mengeksplorasi kemampuan komputer untuk memahami dan memanipulasi teks atau ucapan bahasa alami guna melakukan hal-hal yang bermanfaat. NLP berusaha mencari solusi terhadap masalah dengan memahami bahasa alami manusia, termasuk aturan tata bahasa dan semantik, serta mengubah bahasa tersebut menjadi representasi formal yang dapat diproses oleh komputer.

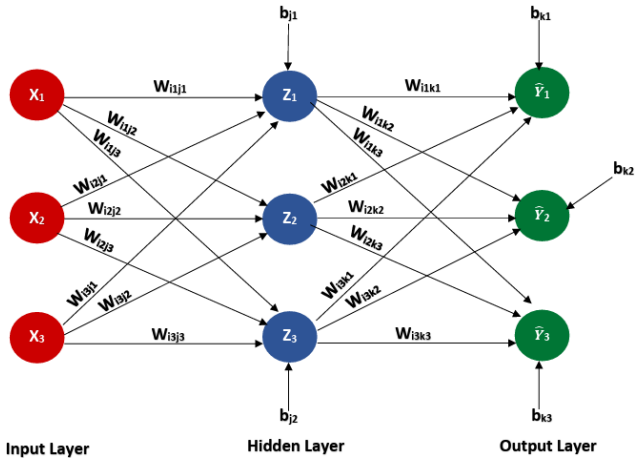
E. Backpropagation Neural Network

Backpropagation Neural Network adalah sebuah jaringan saraf tiruan *Multilayer Perceptron* (MLP) *feed forward* yang menggunakan metode *backpropagation* untuk melatih model. Menurut Tsihrintzis and Jain (2020) Jaringan Syaraf Tiruan (JST) atau *Artificial Neural Network* (ANN) merupakan elemen dasar dari sebagian besar teknik *deep learning*, yang berupaya meniru cara kerja jaringan neuron pada otak manusia. Anandarajan et al. (2019) menambahkan bahwa *Artificial Neural Network* adalah teknik *machine learning* yang belajar dan beradaptasi berdasarkan data yang digunakan dalam model tersebut. Model ini sering disebut sebagai *black box* model karena meskipun memiliki tingkat akurasi yang tinggi, ia sulit dipahami dan diinterpretasikan.

Artificial Neural Network sederhana terdiri dari tiga lapisan yang saling berhubungan, yaitu satu lapisan input, satu lapisan tersembunyi, dan satu lapisan output. Arsitektur *Artificial Neural Network* ini dikenal sebagai arsitektur *Multi-Layer Perceptron* (MLP) atau *Fully Connected Layer*. Lapisan input berfungsi untuk mengumpulkan sinyal atau fitur nilai masukan, lapisan tersembunyi

digunakan untuk melatih bobot, dan lapisan keluaran membentuk lapisan prediksi yang menghasilkan nilai variabel keluaran.

Setiap lapisan pada *Backpropagation Neural Network* terdiri dari sekumpulan unit dasar yang disebut *neuron*, di mana keluaran dari satu lapisan menjadi masukan untuk lapisan berikutnya. Disebut lapisan tersembunyi karena nilai-nilai yang dihasilkan oleh *neuron* di lapisan ini tidak terlihat selama pemrosesan informasi.



Gambar 2.1: Arsitektur Multilayer Perceptron

Arsitektur *Multilayer Perceptron* pada Gambar 2.1 menunjukkan bahwa terdapat 3 *neuron* di *input layer*, 3 *neuron* di *hidden layer*, dan 3 neuron di *output layer*. Struktur ini memungkinkan jaringan saraf untuk belajar dan mengolah informasi secara efisien, menghasilkan prediksi berdasarkan masukan yang diberikan.

Neuron merupakan unit komputasi matematika yang terinspirasi oleh sistem saraf manusia. *Neuron* menerima sekumpulan masukan *X* dari lapisan sebelumnya

yang terhubung dengannya. Setiap masukan tersebut kemudian dikalikan dengan bobot *W* yang sesuai, dan ditambahkan dengan bias *b*. Kombinasi linear antara input dengan bobot yang dijumlahkan dengan bias pada *hidden layer Z* dirumuskan dengan persamaan berikut:

$$Z\_net_j = b_j + \sum_{i=1}^{n} W_{ij} X_i \qquad (2.1)$$

Dimana,

Z_net_j : jumlah input pada *neuron* ke-j pada *hidden layer* z

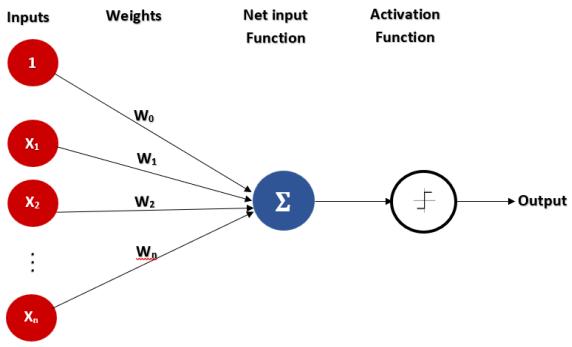
X_i : *input neuron* ke-i pada *input layer*

W_{ij} : bobot penghubung neuron ke-i (*input layer*) dan ke-j (*hidden layer*)

b_j : bias pada *neuron* ke-j pada *hidden layer*

1. Fungsi Aktifasi

Menurut Tsihrintzis and Jain (2020), fungsi aktifasi merupakan persamaan matematika yang ditambahkan pada setiap *neuron* dalam jaringan yang berguna untuk memetakan sinyal masukan menjadi nilai keluaran non-linier. Persamaan ini menghasilkan keputusan bahwa neuron di lapisan berikutnya diaktifkan atau tidak. Gambar 2.2 merupakan operasi dasar ANN dengan neuron yang memiliki fungsi aktivasi menerima sinyal masukan dan menghasilkan keluaran.



Gambar 2.2: Operasi Dasar Artificial Neural Network

Fungsi transformasi non-linier yang disebut sebagai fungsi aktifasi σ diterapkan pada kombinasi linier input *neuron* sehingga menghasilkan output sesuai dengan karakteristik fungsi aktifasi yang digunakan.

$$Output = \sigma\,(W.X + b) \tag{2.2}$$

Terdapat beberapa jenis fungsi aktifasi yang penggunaannya disesuaikan dengan permasalahan yang dihadapi.

a) **Fungsi *Linear***

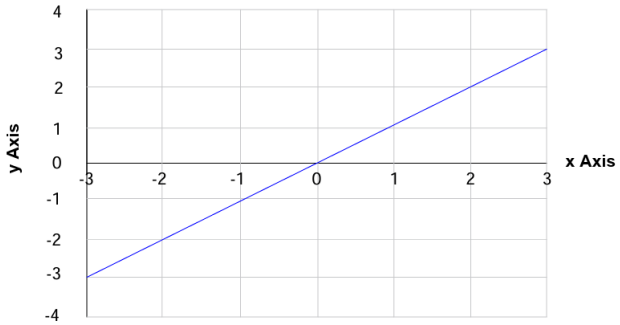
Fungsi aktifasi *linear* menghasilkan output yang nilainya sama dengan hasil kombinasi linear pada inputnya. Jika kombinasi *linear* antara input dengan bobot dijumlahkan dengan bias dinotasikan dengan,

$$x = net_j \tag{2.3}$$

Maka Fungsi aktifasi *linear* dinyatakan

$$F(x) = x \tag{2.4}$$

Gambar 2.3 berikut ini menunjukkan grafik fungsi aktifasi *Linear*.

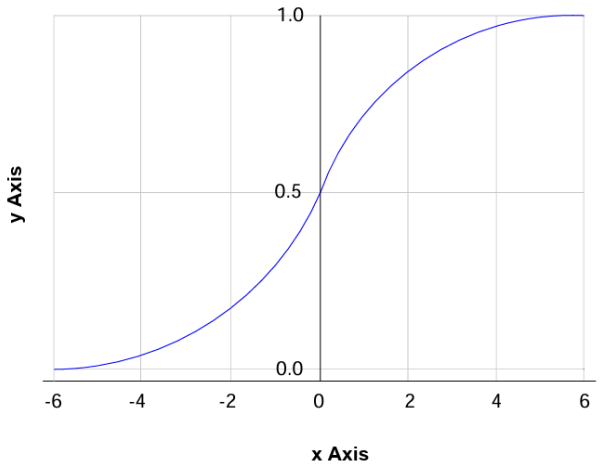


Gambar 2.3: Fungsi Aktifasi Linear

b) **Fungsi *Sigmoid***

Fungsi aktifasi *Sigmoid* melakukan normalisasi output pada rentang 0 hingga 1. Fungsi aktifasi *Sigmoid* ini memiliki kurva berbentuk S. Gambar 2.4

berikut ini menunjukkan grafik fungsi aktifasi *Sigmoid*.



Gambar 2.4: Fungsi Aktifasi *Sigmoid*

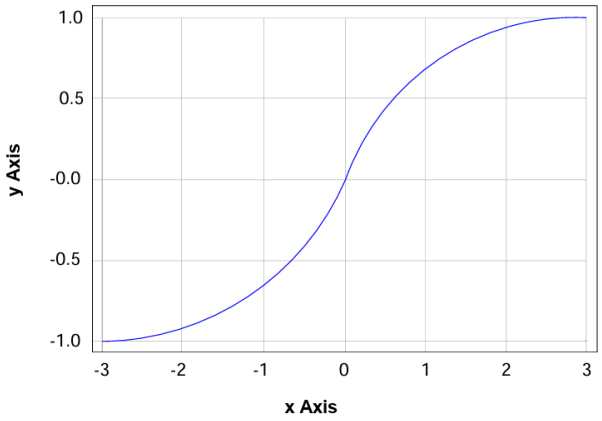
Persamaan fungsi aktifasi *Sigmoid* adalah

$$F(x) = 1\ /1\ + \exp(x) \tag{2.5}$$

Fungsi aktifasi ini memiliki gradien nilai output yang *smooth*. Kelemahan fungsi aktifasi ini adalah adanya permasalahan *vanishing gradient*. Yaitu, nilai gradien yang sangat kecil sehingga hampir tidak ada gradien pada output nol dan satu. Kelemahan lain dari fungsi aktifasi ini adalah memerlukan waktu yang lama untuk mencapai bobot yang konvergen dan optimasi yang tidak mudah dicapai dikarenakan *Sigmoid* tidak *zero-centered*.

c) Fungsi *Tanh*

Fungsi aktifasi *Tanh* melakukan normalisasi output pada rentang -1 hingga 1. Kurva fungsi aktifasi *Tanh* berbentuk S. Gambar 2.5 berikut ini menunjukkan grafik fungsi aktifasi *Tanh*.



Gambar 2.5: Fungsi Aktifasi Tanh

Persamaan fungsi aktifasi *Tanh* adalah

$$F(x) = 1 - \exp(-2x)\ /1\ + \exp(-2x) \qquad (2.6)$$

Fungsi aktifasi ini mudah di-optimasi karena output *tanh zero-centered* dan memiliki gradien nilai output yang *smooth*. Namun, sama seperti fungsi aktifasi *Sigmoid*, fungsi aktifasi ini memiliki permasalahan *vanishing gradient* dan memerlukan waktu yang lama untuk mencapai bobot yang konvergen.

d) Fungsi *Softmax*

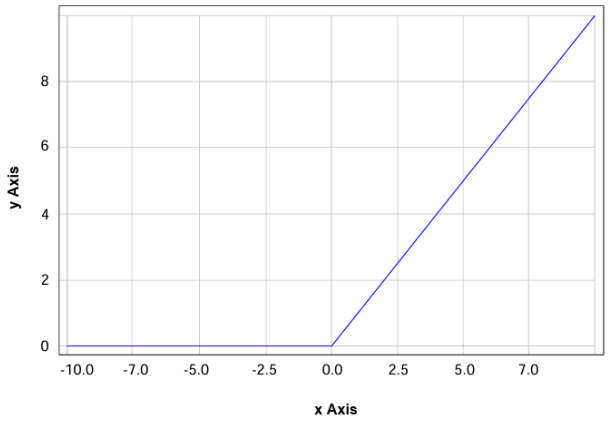
Fungsi aktifasi *Softmax* digunakan untuk menangani *multiclass*. Fungsi aktifasi ini menghitung probabilitas setiap kelas, memiliki rentang nilai antara 0 hingga 1 dan jumlah probabilitas seluruh kelas adalah 1. Kelas target merupakan kelas yang memiliki probabilitas tertinggi. Persamaan fungsi aktifasi *Softmax* adalah

$$F(x) = \frac{\exp(x_i)}{\sum \exp(x_i)} \qquad (2.7)$$

e) Fungsi *Rectified Linear Unit (ReLu)*

Fungsi Aktifasi *ReLU* memiliki output bernilai 0 sampai positif tak hingga.

Gambar 2.6 berikut ini menunjukkan grafik fungsi aktifasi *ReLu*.



Gambar 2.6: Fungsi Aktifasi *ReLU*

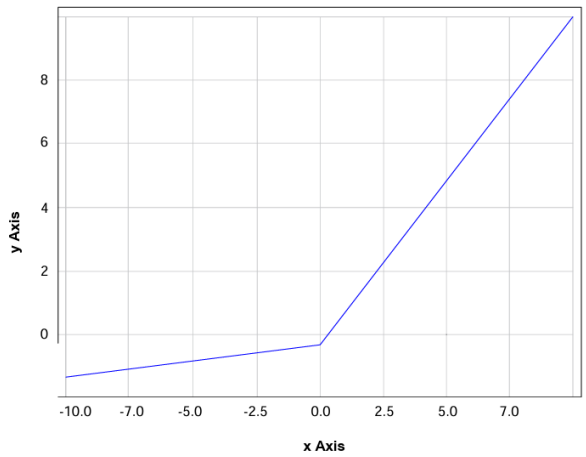
Persamaan fungsi aktifasi *ReLU* adalah

$$R(x) = \max(0, x) \tag{2.8}$$

Fungsi aktifasi *ReLU* mampu mengatasi *vanishing gradient* yang dialami fungsi aktifasi *Sigmoid* dan *Tanh*. *ReLU* sangat efisien dalam proses *back-propagation* untuk mencapai bobot yang konvergen dengan cepat. Namun, apabila input *neuron* di bawah nol akan menyebabkan *neuron* tersebut mati (*Dying ReLU*) sehingga proses *back-propagation* tidak efisien.

f) Fungsi *Leaky ReLU*

Fungsi Aktifasi *Leaky ReLU* memiliki output bernilai minus tak hingga sampai dengan positif tak hingga. Gambar 2.7 berikut ini menunjukkan grafik fungsi aktifasi *Leaky ReLu*.



Gambar 2.7: Fungsi Aktifasi *Leaky ReLU*

Persamaan fungsi aktifasi *Leaky ReLU* adalah

$$F(x) = \max(0.1 * x, x) \tag{2.9}$$

Fungsi aktifasi *Leaky ReLU* mampu mengatasi *vanishing gradient* dan mencegah *Dying ReLU*. *Leaky* ReLU cepat dalam mencapai bobot yang konvergen. Namun, fungsi aktifasi *Leaky ReLU* tidak konsisten dalam melakukan prediksi disebabkan hasil input negatif yang tidak efisien.

2. Proses *Training*

Model yang telah dirancang dapat belajar dari data yang diberikan melalui proses pelatihan. Seluruh proses pelatihan pada Gambar 2.1 mengikuti tahapan berikut ini:

1. Inisialisasi model; inisialisasi parameter/bobot secara acak menggunakan distribusi normal seragam atau standar.
2. *Feed forward*; sinyal masukan yang diterima unit *input layer* diteruskan melalui hidden layer yang setiap unitnya memiliki fungsi aktifasi yang keluarannya diteruskan menuju *output layer*. Setiap unit *input layer* (X_i, i = 1, 2, …., n)

menerima sinyal input dan meneruskan sinyal tersebut ke seluruh unit pada *hidden layer*. Setiap unit *hidden layer* (Z_j, j = 1, 2, …, p) menjumlahkan perkalian X_i dan bobot W_{ij} ditambah dengan bias b_j.

$$Z\_net_j = b_j + \sum_{i=1}^{n} X_i W_{ij} \tag{2.10}$$

Sinyal output unit tersembunyi diperoleh dengan menerapkan fungsi aktifasi.

$$Z_j = f(Z\_net_j) \tag{2.11}$$

Misalkan fungsi aktifasi yang digunakan adalah *ReLU*.

$$relu = \max(0, x) \tag{2.12}$$

Selanjutnya sinyal output dari *hidden layer* ini dikirim ke seluruh unit pada *output layer*. Setiap unit pada *output layer* ($\hat{y}_k$, k = 1, 2, …, q) menjumlahkan perkalian Z_j dan bobot W_{jk} ditambah dengan bias b_k.

$$\hat{y}\_net_k = b_k + \sum_{j=1}^{p} Z_j W_{jk} \tag{2.13}$$

Sinyal output unit pada lapisan keluaran diperoleh dengan menerapkan fungsi aktifasi.

$$\hat{y}_k = f(\hat{y}\_net_k) \tag{2.14}$$

Misalkan fungsi aktifasi yang digunakan adalah *softmax*. Maka,

$$\hat{y}_k = \frac{e^{\hat{y}\_net_k}}{\sum_{k=1}^{q} e^{\hat{y}\_net_k}} \tag{2.15}$$

3. *Loss Function*; merupakan fungsi yang membandingkan nilai output target dengan prediksi. Model jaringan syaraf mempelajari hubungan antara data pelatihan dan output selanjutnya memperbarui *hyperparameter*-nya berupa bobot dan bias sampai menghasilkan nilai prediksi yang mendekati nilai target.

Misalkan *Loss function* yang digunakan untuk klasifikasi lebih dari dua kelas adalah *Categorical Cross-Entropy*.

$$E = f(\hat{y}_1, \hat{y}_2, \hat{y}_3, \ldots \hat{y}_k) \tag{2.16}$$

$$E = -\sum_{k=1}^{q} y_k \cdot log(\hat{y}_k) \tag{2.17}$$

4. *Backpropagation*; merupakan proses men-distribusikan kesalahan total dihitung kembali ke langkah awal (*backward*) menggunakan *gradient descent*. Metode ini diterapkan untuk mengoptimalkan bobot dan bias yang akhirnya akan mengurangi kesalahan total. *Gradient descent* merupakan konsep matematika untuk memperoleh posisi global minima atau lokal minima dengan menghitung turunan fungsi kerugian sebuah model. *Gradient Descent* erat kaitannya *learning rate* (α). *Learning rate* yang terlalu besar akan mengakibatkan kehilangan minima, sedangkan *learning rate* yang terlalu kecil mengakibatkan proses pelatihan terlalu lambat. *Learning rate* sebaiknya berada pada rentang 10^{-1} hingga $10^{3.}$

Memperbarui bobot *hidden layer–output layer* (W_{jk}) dan *bias output layer* (b_k)

Informasi *error* setiap unit *output layer* (δ_k) adalah:

$$\delta_k = \frac{\partial E}{\hat{y}_k} \, f'(\hat{y}\_net_k) \tag{2.18}$$

Turunan *error* (E) terhadap output prediksi ($\hat{y}$) adalah

$$\frac{\partial E}{(\hat{y}_1,\hat{y}_2,\hat{y}_3,\ldots\hat{y}_k)} = \begin{bmatrix} \frac{\partial E}{\partial \hat{y}_1} \\ \frac{\partial E}{\partial \hat{y}_2} \\ \frac{\partial E}{\partial \hat{y}_3} \\ \cdots \\ \frac{\partial E}{\partial \hat{y}_k} \end{bmatrix} = \begin{bmatrix} -\frac{y_1}{\hat{y}_1} \\ -\frac{y_2}{\hat{y}_2} \\ -\frac{y_3}{\hat{y}_3} \\ \cdots \\ -\frac{y_k}{\hat{y}_k} \end{bmatrix} \quad (2.19)$$

Sehingga,

$$\delta_k = -\frac{y_k}{\hat{y}_k} \, \mathrm{f}'\left(\hat{y}_{\_net_k}\right) \quad (2.20)$$

$\mathrm{f}'\left(\hat{y}_{\_net}\right)$ adalah turunan output prediksi ($\hat{y}$) terhadap input prediksi ($\hat{y}_{\_net}$) yang merupakan turunan dari fungsi *Softmax*. Sehingga,

$$\frac{\partial \hat{y}_1}{\partial \hat{y}_{\_net_1}} = \frac{e^{y\_net1} \cdot \left(e^{y\_net2} + e^{y\_net3} + \cdots + e^{y\_netk}\right)}{\left(e^{y\_net1} + e^{y\_net2} + e^{y\_net3} + \cdots + e^{y\_netk}\right)^2} \quad (2.21)$$

$$\frac{\partial \hat{y}_2}{\partial \hat{y}_{\_net_2}} = \frac{e^{y\_net2} \cdot \left(e^{y\_net1} + e^{y\_net3} + \cdots + e^{y\_netk}\right)}{\left(e^{y\_net1} + e^{y\_net2} + e^{y\_net3} + \cdots + e^{y\_netk}\right)^2} \quad (2.22)$$

Turunan input prediksi ($\hat{y}_{\_net}$) terhadap bobot antara *hidden layer* dan *output layer* (W_{jk}) adalah:

$$\frac{\partial \hat{y}_{\_net_1}}{\partial W_{j1k1}} = \frac{\partial\left(\left(Z_1 * W_{j1k1}\right) + \left(Z_2 * W_{j2k1}\right) + \left(Z_3 * W_{j3k1}\right) + b_1\right)}{\partial W_{j1k1}} \quad (2.23)$$

$$\frac{\partial \hat{y}_{\_net_1}}{\partial W_{j1k1}} = Z_1 \quad (2.24)$$

Menghitung gradien bobot antara *hidden layer* dan *output layer* (δW_{jk}) menggunakan aturan rantai.

$$\delta W_{jk} = \frac{\partial E}{\partial W_{jk}} = \frac{\partial E}{\partial \hat{y}_k} * \frac{\partial \hat{y}_k}{\partial \hat{y}_{ink}} * \frac{\partial \hat{y}_{in}}{\partial W_{jk}} \quad (2.25)$$

$$\delta W_{jk} = \frac{\partial E}{\partial W_{jk}} = \delta_k * Z_j \quad (2.26)$$

Memperbarui bobot W_{jk}

$$W_{jk}(\text{baru}) = W_{jk}(\text{lama}) - \alpha \delta_k Z_j \quad (2.27)$$

Memperbarui bias b_{k}

$$b_{\mathrm{k}}(\mathrm{baru}) = b_{\mathrm{k}}(\mathrm{lama}) - \ \alpha\delta_k \tag{2.28}$$

Memperbarui bobot *input layer–hidden layer* (W_{ij}) dan bias *hidden layer* (b_j)

Turunan *error* (E) terhadap keluaran *hidden layer* (Z_j) adalah:

$$\frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial Z_j} * \frac{\partial Z_j}{\partial Z_{inj}} * \frac{\partial Z_{inj}}{\partial W_{ij}} \tag{2.29}$$

Dimana:

- $\frac{\partial E}{\partial Z_j}$, setiap keluaran unit *hidden layer* (Z_j, j = 1, 2, …, p) merupakan penjumlahan dari perkalian informasi *error* (δ_k) dengan bobot-bobot yang menghubungkan *hidden layer* dan *output layer* (W_{jk}).

$$\delta_{\_net_j} = \sum_{k=1}^{m} \delta_k \, W_{jk} \tag{2.30}$$

- $\frac{\partial Z_j}{\partial Z_{\_netj}}$, merupakan turunan dari fungsi aktifasi *ReLU*.

$$\mathrm{f}'\left(z_{\_net_j}\right) \Rightarrow \quad if\ z_{\_net_j} > 0, \frac{\partial(\mathrm{relu})}{\partial Z_{\_netj}} = 1$$

$$\text{selain itu, } \frac{\partial(\mathrm{relu})}{\partial Z_{\_netj}} = 0 \tag{2.31}$$

- $\frac{\partial Z_{\_netj}}{\partial W_{ij}} = X_{\mathrm{i}}$

Informasi *error* setiap unit *output layer* (δ_k) diperoleh dari perkalian $\delta_{\_net_j}$ dan turunan dari fungsi aktifasi *ReLU*.

$$\delta_j = \delta_{in_j} \mathrm{f}'\left(z_{in_j}\right) \tag{2.32}$$

Memperbarui bobot W_{ij}

$$W_{ij}(\mathrm{baru}) = W_{ij}(\mathrm{lama}) - \ \alpha\delta_j X_{\mathrm{i}} \tag{2.33}$$

Memperbarui bias b_j

$$b_j(\text{baru}) = \text{b}_j(\text{lama}) - \ \alpha\delta_j \tag{2.34}$$

5. Iterasi sampai konvergen; mengulangi pelatihan dari angka 2 sampai dengan 4 sampai nilai parameter tercapai pada rata-rata *error training* rendah atau *error* mencapai batas toleransi. Pada keadaan itu iterasi akan terhenti. Iterasi juga dapat terhenti jika mencapai jumlah iterasi yang telah ditetapkan sebelumnya sekalipun belum konvergen.

F. Data Tidak Seimbang

Kondisi nyata di lapangan sering menunjukkan adanya ketidakseimbangan dalam kumpulan data, di mana kelas dengan proporsi besar disebut kelas mayoritas, sedangkan kelas dengan proporsi lebih kecil disebut kelas minoritas. Ketidakseimbangan ini tidak menjadi masalah jika selisihnya kecil, tetapi bisa menjadi tantangan serius jika ada satu atau lebih kelas yang sangat jarang. Dalam situasi seperti ini, banyak model klasifikasi kesulitan dalam mengidentifikasi kelas minoritas dengan akurat, yang dapat mengakibatkan masalah dalam pelatihan dan evaluasi model.

Menurut Campesato (2021), algoritma klasifikasi cenderung tidak berfungsi dengan baik pada data yang tidak seimbang. Salah satu pendekatan untuk menangani masalah ini adalah dengan melakukan pengambilan sampel ulang pada dataset, yang lebih dikenal sebagai *resampling dataset*. Metode ini bertujuan untuk menciptakan keseimbangan antara kelas-kelas dalam dataset sehingga model klasifikasi dapat belajar dan beradaptasi dengan lebih efektif terhadap semua kelas yang ada, baik mayoritas maupun minoritas.

1. *Random Undersampling*

Random undersampling atau *random subsampling* adalah teknik pengambilan sampel yang dilakukan secara acak dengan cara menghapus atau mengurangi jumlah sampel dari kelas mayoritas. Tujuannya adalah untuk menyeimbangkan jumlah sampel antara kelas mayoritas dan kelas minoritas. Namun, meskipun metode ini efektif dalam mengatasi ketidakseimbangan data, penghapusan sampel secara acak dapat menyebabkan kehilangan informasi berharga yang berpotensi membuat model mengalami *underfitting*, di mana model tidak dapat menangkap pola yang ada dalam data.

Teknik *random undersampling* ini biasanya cocok diterapkan pada dataset dengan jumlah lebih dari 10 ribu sampel. Dalam situasi ini, meskipun penghapusan sebagian data mungkin tampak merugikan, teknik ini tetap dapat membantu menciptakan model yang lebih baik jika dilakukan dengan hati-hati, mengingat bahwa penghapusan yang tidak tepat bisa mengurangi akurasi dan kemampuan generalisasi model.

2. *Random Oversampling*

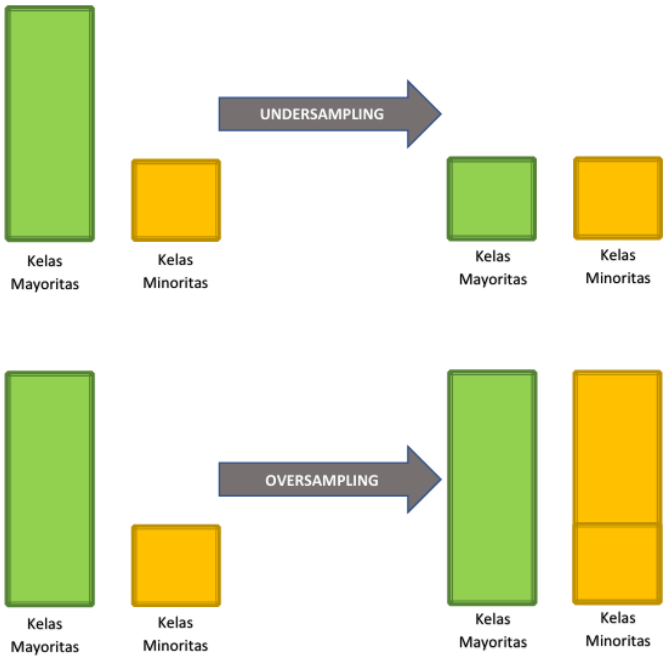
Random oversampling adalah teknik pengambilan sampel yang dilakukan secara acak untuk menyeimbangkan kumpulan data dengan cara menduplikasi sampel pada kelas minoritas. Dengan cara ini, jumlah sampel di kelas minoritas dapat ditingkatkan sehingga seimbang dengan kelas mayoritas, yang memungkinkan model untuk belajar lebih baik dari data minoritas.

Meskipun teknik ini dapat menghasilkan model yang lebih kompetitif, ada beberapa risiko yang perlu diperhatikan. Salah satunya adalah kemungkinan terjadinya *overfitting*, di mana model menjadi terlalu sesuai dengan data pelatihan

dan kehilangan kemampuan untuk generalisasi pada data baru. Selain itu, proses duplikasi sampel juga dapat menyebabkan PC *overheat* akibat peningkatan komputasi yang diperlukan untuk memproses data yang lebih besar.

Random oversampling ini lebih cocok diterapkan pada dataset dengan jumlah kurang dari 10 ribu sampel, karena pada dataset yang lebih besar, teknik ini mungkin tidak efektif dan dapat memperlambat proses pelatihan tanpa memberikan peningkatan yang signifikan dalam kinerja model.

Gambar 2.8 merupakan ilustrasi dari proses *Undersampling* dan *Oversampling*.



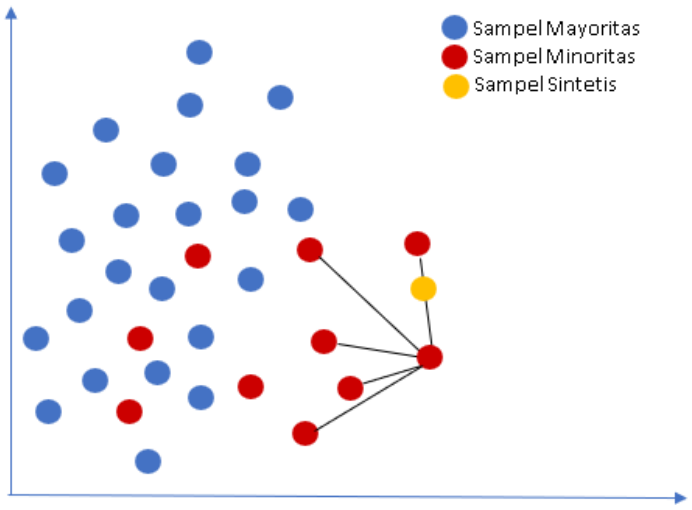
Gambar 2.8: Ilustrasi Undersampling dan Oversampling

3. SMOTE

Penerapan teknik SMOTE sebagai upaya untuk menyeimbangkan jumlah data antar kelas. SMOTE adalah pendekatan *oversampling* yang bekerja dengan

cara meningkatkan jumlah data pada kelas minoritas melalui sintesis sampel baru, sehingga jumlah data kelas minoritas tersebut sama dengan jumlah data kelas mayoritas sebelum diklasifikasikan.

Algoritma SMOTE bekerja dengan cara menciptakan sampel sintetis untuk tetangga terdekat yang dipilih, dengan menghitung selisih vektor antara sampel referensi dan tetangga tersebut (Firmansyah et al., 2023). SMOTE memilih beberapa sampel dari kelas minoritas. Untuk setiap sampel yang dipilih, algoritma mencari k tetangga terdekat (*neighbors*) dari sampel tersebut. Sampel sintesis dibuat dengan mengambil titik di sepanjang garis penghubung antara sampel awal dan tetangga terdekat yang terpilih secara acak. Jumlah k biasanya ditentukan berdasarkan berapa banyak *oversampling* yang ingin dilakukan dan dipilih secara acak (Chawla et al., 2002).



Gambar 2.9: Ilustrasi SMOTE (Zhang et al., 2023)

Pada ruang fitur yang ditunjukkan pada Gambar 2.9, diasumsikan bahwa terdapat sampel utama *x* dari kelas minoritas yang memiliki K=5 sampel tetangga

terdekat. Sampel tetangga terdekat x' dipilih secara acak, kemudian sampel baru x^{baru} dihasilkan pada posisi acak pada garis koneksi antara sampel utama x dan sampel tetangga terdekat x'. Sintesis sampel baru x^{baru}dihitung seperti yang ditunjukkan pada Persamaan (2.35)

$$x^{baru} = x + rand(0,1) * (x' - x) \quad (2.35)$$

Dimana

x : Vektor dari sampel minoritas asli yang dipilih

x' : Vektor dari sampel minoritas tetangga terdekat yang dipilih secara acak

x^{baru} : Vektor sampel baru sintetis

$rand(0,1)$: Angka acak yang diambil dari rentang antara 0 dan 1

$(x' - x)$: Selisih antara kedua vektor x dan x'

Proses pembuatan sampel sintetis diulang untuk setiap sampel di kelas minoritas sampai jumlah sampel pada kelas minoritas menjadi setara dengan jumlah sampel di kelas mayoritas. Dengan kata lain, SMOTE akan terus menghasilkan sampel sintetis hingga kelas minoritas memiliki cukup sampel untuk menyeimbangkan distribusi kelas. Dengan menggunakan SMOTE, model *machine learning* diharapkan dapat mengenali pola dari kelas minoritas dengan lebih baik, sehingga meningkatkan kinerja dan akurasi model dalam memprediksi semua kelas.

G. TF-IDF

Teks sebagai data bertipe string tidak dapat diolah secara langsung menggunakan algoritma pembelajaran. Data bertipe ini harus diubah atau direpresentasikan terlebih dahulu menjadi bentuk angka atau vektor. Proses ini dikenal sebagai ekstraksi fitur.

Menurut Campesato (2021), ekstraksi kata kunci sebuah dokumen merupakan proses yang paling signifikan. *Term Frequency - Inverse Document Frequency* (TF-IDF) adalah salah satu teknik untuk melakukan ekstraksi kata kunci dengan menghitung bobot sebuah kata dalam korpus. Pembobotan TF menunjukkan seberapa penting sebuah kata tersebut dalam sebuah dokumen. Sedangkan pembobotan IDF menunjukkan seberapa sering sebuah kata tersebut dalam seluruh dokumen. Sehingga, TF-IDF merupakan kombinasi antara *term frequency* (tf) dan *inverse document frequency* (idf) dengan mengalikan keduanya.

$$\mathrm{w}_{ij} = tf_{ij} * idf_i \tag{2.36}$$

$$idf_i = \log\left(\frac{N}{df_i}\right) + 1 \tag{2.37}$$

Dimana

w_{ij} : bobot kata *i* pada dokumen *j*

tf_{ij} : frekuensi kata *i* pada dokumen *j*

idf_i : *inverse document frequency*

df_i : frekuensi dokumen yang mengandung kata *i*

i : kata 1, 2, … dst

j : dokumen 1, 2, …, N

Sehingga,

$$\mathrm{w}_{ij} = tf_{ij} * (\log\left(\frac{N}{df_i}\right) + 1) \tag{2.38}$$

Persamaan 2.38 menggunakan *linear* TF dan *non-smooth* IDF. Dengan mengacu pada *library Scikitlearn*, pembobotan TF dan IDF memiliki variansi seperti pada tabel berikut ini.

Tabel 2.1: Rumus Pembobotan TF-IDF

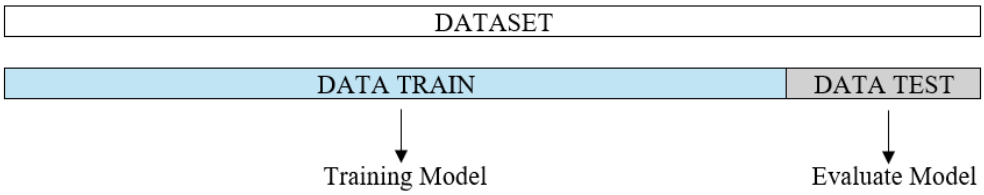
Pembobotan TF		**Pembobotan IDF**	
linear	tf	*non-smooth*	$\log\ (\frac{N}{df})+1$
sublinear	$1+\log\,(tf)$	*smooth*	$\log\ \left(\frac{N+1}{df+1}\right)+1$

H. Validasi Model

Validasi model dilakukan untuk pengujian performa model *machine learning* pada data yang belum pernah dikenal sebelumnya. Validasi model berguna untuk memastikan bahwa model dapat menghasilkan prediksi yang akurat dan stabil tidak hanya pada data training tetapi juga pada data baru yang belum dikenal model. Validasi model dapat dilakukan dengan pemisahan data latih dan data testing (*train-test split*) atau dengan validasi silang (*cross-validation*).

1. Train-Test Split

Train-test split validation merupakan metode validasi model sederhana dan sering digunakan. Dataset dibagi menjadi dua bagian yaitu, data *training* dan data *testing*. Umumnya, perbandingan antara data *training* dan data *testing* adalah 70:30, 75:25, 80:20 atau 90:10. Gambar 2.10 menunjukkan pembagian dataset menjadi data *training* dan data *testing*.

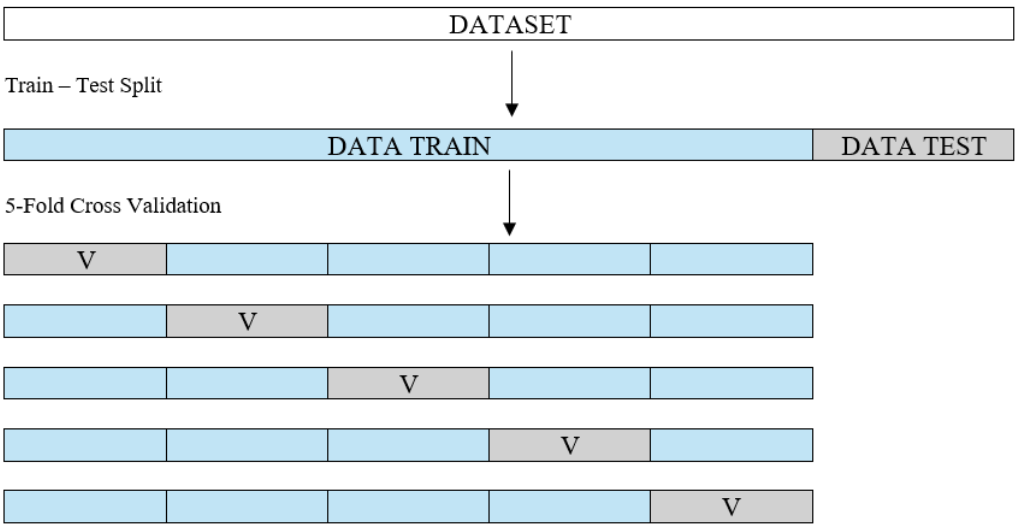


Gambar 2.10: Pembagian Dataset dengan *Train-Test Split*

Data *training* digunakan untuk melatih model, sedangkan data *testing* digunakan untuk menguji model. Metode ini memiliki kekurangan, terutama jika jumlah dataset yang digunakan sedikit dan distribusi kelas tidak seimbang.

2. *Cross-Validation*

Cross-validation merupakan metode validasi model yang mampu menangani kekurangan pada validasi model *train-test split*. Prinsip *cross-validation* membagi data menjadi beberapa bagian atau *subset*. Proses belajar model menggunakan sebagian besar subset sebagai data *training* dan satu subset sebagai data *testing*. Proses pembelajaran diulangi sampai semua subset pernah berperan sebagai data *testing*. Gambar 2.11 menunjukkan contoh pelatihan pada data *training* menggunakan *cross-validation*.



Gambar 2.11: Lima *Fold Cross-Validation*

Cross-validation pada data *training* dibagi menjadi beberapa bagian atau *subset*. Proses belajar model menggunakan sebagian besar subset sebagai data

training dan satu subset sebagai data validasi. Proses belajar diulangi sampai semua *subset* pernah berperan sebagai data validasi.

Cross-validation atau validasi silang merupakan suatu pendekatan dengan cara mengambil sampel ulang statistik untuk mengevaluasi performa model pembelajaran mesin pada data yang tidak dilihat guna mendapatkan performa yang akurat dan konsisten. *Cross-validation* juga berguna untuk menjadikan *hyperparameter* lebih optimal.

I. Metrik Evaluasi

Ketepatan klasifikasi sentimen yang telah dilakukan dapat diketahui melalui pengukuran. Menurut pengujian (Khare et al., 2022), *Confusion Matrix* adalah teknik untuk mengukur kinerja dalam klasifikasi pembelajaran mesin yang diwakili oleh semacam tabel untuk mengetahui kinerja dari model klasifikasi pada data. Tabel ini menyajikan kelas aktual dan prediksi berdasarkan model analisis klasifikasi. *Confusion Matrix* merupakan cara cepat untuk membandingkan label yang diprediksi oleh model dan label sebenarnya yang seharusnya diprediksi. *Confusion Matrix* untuk klasifikasi biner ditunjukkan pada Tabel 2.2.

Tabel 2.2 : *Confusion Matrix* untuk Klasifikasi Biner

Kelas Aktual	**Di-klasifikasi-kan sebagai positif**	**Di-klasifikasi-kan sebagai negatif**
Positif	*True Positive* (TP)	*False Negative* (FN)
Negatif	*False Positive* (FP)	*True Negative* (TN)

Berikut ini beberapa nilai yang terdapat pada Confusion Matrix:

1. *True Positive* (TP) adalah nilai-nilai yang positif dalam kenyataan dan

diprediksi positif juga,

2. *False Positive* (FP) adalah nilai yang negatif dalam kenyataan tetapi diprediksi sebagai positif,
3. *False Negative* (FN) adalah nilai yang positif dalam kenyataan tetapi diprediksi sebagai negatif, dan
4. *True Negative* (TN) adalah nilai yang negatif dalam kenyataan dan diprediksi negatif juga.

Penilaian ketepatan klasifikasi *multiclass* dihitung untuk masing-masing kelas. Penilaian ketepatan klasifikasi untuk *multiclass* berdasar pada Tabel 2.2 untuk banyak kelas, maka kelas C_i, TP_i adalah *True Positive* untuk kelas C_i, FN_i adalah *False Negative* untuk kelas C_i, TN_i adalah *True Negative* untuk kelas C_i, FP_i adalah *False Positive* untuk kelas C_i $accuracy_i$ adalah *accuracy* untuk kelas C_i, $precision_i$ adalah *precision* untuk kelas C_i, $recall_i$ adalah *recall* untuk kelas C_i.

Sedangkan ketepatan klasifikasi secara keseluruhan dapat dinilai melalui dua cara, yaitu rata-rata makro (*macro-averaging*) dan rata-rata mikro (*micro-averaging*). *Rata-rata makro* dihitung dari rata-rata sebuah pengukuran yang dihitung dari semua kelas seperti ditunjukkan persamaan dengan indeks M pada Tabel 2.3. Rata-rata makro memperlakukan setara untuk semua kelas. Sedangkan rata-rata mikro dihitung dari jumlah kumulatif masing-masing TP, FN, TN, dan FP, kemudian dilakukan perhitungan performa pengukuran seperti ditunjukkan persamaan dengan indeks μ pada Tabel 2.3. Rata-rata mikro mengutamakan kelas yang lebih besar.

Tabel 2.3: Pengukuran Performa Model untuk *Multiclass*

Ukuran	Rumus
Accuracy rata-rata	$\frac{\sum_{i=1}^{n} \frac{tp_i + tn_i}{tp_i + fn_i + fp_i + tn_i}}{n}$
Error rate	$\frac{\sum_{i=1}^{n} \frac{fp_i + fn_i}{tp_i + fn_i + fp_i + tn_i}}{n}$
Precision $_{\mu}$	$\frac{\sum_{i=1}^{n} tp_i}{\sum_{i=1}^{n} (tp_i + fp_i)}$
Recall $_{\mu}$	$\frac{\sum_{i=1}^{n} tp_i}{\sum_{i=1}^{n} (tp_i + fn_i)}$
Fscore $_{\mu}$	$2 * \frac{Precision_{\mu}\, Recall_{\mu}}{Precision_{\mu} + Recall_{\mu}}$
Precision $_{M}$	$\frac{\sum_{i=1}^{n} \frac{tp_i}{tp_i + fp_i}}{n}$
Recall $_{M}$	$\frac{\sum_{i=1}^{n} \frac{tp_i}{tp_i + fn_i}}{n}$
Fscore $_{M}$	$2 * \frac{Precision_{M}\, Recall_{M}}{Precision_{M} + Recall_{M}}$

J. Penelitian Sebelumnya

Penelitian penerapan *Backpropagation Neural Network* dan SMOTE pada analisis sentimen ulasan pengguna *MyXL* di *Google Play Store* ini didasari oleh beberapa penelitian yang telah dilakukan sebelumnya.

Analisis sentimen ulasan pengguna aplikasi *MyXL* menggunakan metode *Support Vector Machine* (Audiansyah et al., 2022). Penelitian ini mengklasifikasikan dua kelas sentimen dengan data seimbang berjumlah 160 positif dan 160 negatif. Uji coba dilakukan menggunakan 5 *cross-validation* dan *kernel linear* menghasilkan nilai rerata *accuracy*, *precision*, *recall*, dan *f-measure* 88%.

Analisis sentimen komentar pengguna Gojek dan *Grab* di *Google Play* menggunakan algoritma *Naïve Bayes* dan *Support Vector Machine* dengan

menerapkan teknik SMOTE (Hermanto et al., 2020). Penelitian ini menggunakan 2,160 data komentar pengguna dengan komentar pengguna Gojek berjumlah 1,160 dan komentar pengguna *Grab* berjumlah 1,000. Komentar pengguna diklasifikasikan negatif dan positif pada masing-masing aplikasi. Penerapan teknik SMOTE pada model *Support Vector Machine* memperoleh metrik evaluasi tertinggi pada kedua aplikasi Gojek dan *Grab*. Aplikasi Gojek memperoleh akurasi 81,09% dan nilai AUC 0,922, sedangkan aplikasi *Grab* memperoleh akurasi 73,20% dan nilai AUC 0,848.

Analisis sentimen yang membandingkan metode *Logistic Regression*, *Multinomial Naïve Bayes*, SVM, dan K-NN pada *review* aplikasi Gojek di *Google Play Store* (Maulana et al., 2023). Penelitian ini menggunakan 2,000 sampel yang diambil 717 data sentimen negatif dan 705 data sentimen positif, sentimen netral tidak digunakan dalam perbandingan. Metode *Logistic Regression* memperoleh performa tertinggi dengan skor accuracy 82,45%, recall 82,49%, precision 82,45%, dan f1-score 82.43%.

Analisis ulasan pengguna aplikasi PeduliLindungi di *Google Play Store* menggunakan algoritma *Support Vector Machine* dan *Naïve Bayes* berdasarkan *Particle Swarm Optimization* (Mustopa et al., 2020). Penelitian ini menggunakan 1,364 data ulasan pengguna yang kemudian dikelompokkan berdasarkan kategori positif dan negatif. *Support Vector Machine* berdasarkan *Particle Swarm Optimization* memperoleh metrik evaluasi lebih tinggi dari *Naïve Bayes* berdasarkan *Particle Swarm Optimization* dengan nilai akurasi 93.0% dan nilai AUC 0,977.

Analisis sentimen ulasan pengguna aplikasi *Mypertamina* pada *Google Playstore* menggunakan metode *Naïve Bayes* (Gilbert et al., 2023). Penelitian ini menggunakan 3,948 data ulasan pengguna yang kemudian dikelompokkan berdasarkan kategori positif dan negatif. Metode *Naive Bayes* memperoleh performa tertinggi dengan skor accuracy 91%, precision 92%, recall 100%.

Analisis sentimen ulasan aplikasi *Amazon Shopping* Di *Google Play Store* menggunakan *Naive Bayes Classifier* (Hasibuan and Heriyanto, 2022). Penelitian ini menggunakan 991 ulasan bersentimen positif dan 1156 bersentimen negatif. Pengklasifikasi *Naive Bayes* yang digunakan adalah *BernoulliNB, ComplementNB, GaussianNB,* dan *MultinomialNB*. Pengklasifikasi *MultinomialNB* memperoleh performa tertinggi dengan skor *precision* 78.82%, *recall* 85.90% dan *f1-score* 82.21%

Analisis sentimen ulasan pengguna aplikasi Dana di *Google Play* menggunakan metode *Support Vector Machine* (Muhammad et al., 2022). Metode SVM dengan seleksi fitur *chi square* diperoleh akurasi sebesar 89,41%, presisi sebesar 93,29%, dan *recall* sebesar 90,76%.

Analisis sentimen pengguna aplikasi JMO (Jamsostek Mobile) pada *Google Play Store* menggunakan metode *Naive Bayes* (Rizaldi et al., 2023). Penelitian ini terdiri dari sentimen positif sebanyak 1,472 dan sentimen negatif sebanyak 3,528. Klasifikasi menggunakan metode naive bayes menghasilkan akurasi 95%, *precision* 91% dan (*recall*) 90%.

Penelitian-penelitian tersebut di atas disajikan dalam bentuk tabel pada Tabel 2.4.

Tabel 2.4: Rangkuman Penelitian Sebelumnya

No	Peneliti	Judul	Fitur	Daya Beda
1	Dimas Diandra Audiansyah, Dian Eka Ratnawati, Buce Trias Hanggara (2022)	Analisis Sentimen Aplikasi *MyXL* menggunakan Metode *Support Vector Machine* berdasarkan Ulasan Pengguna di *Google Play Store*	▪ analisis sentimen ▪ Support Vector Machine ▪ ulasan pengguna MyXL di Google Play Store ▪ sentimen positif (160) dan negatif (160)	▪ analisis sentimen ▪ BackPropagation Neural Network ▪ SMOTE ▪ ulasan pengguna MyXL di Google Play Store ▪ sentimen negatif (613), netral (226) dan positif (161)
2	Hermanto, Antonius Yadi Kuntoro, Taufik Asra, Eri Bayu Pratama, Lasman Effendi, and Ridatu Ocanitra (2020)	*Gojek and Grab User Sentiment Analysis on Google Play Using Naive Bayes Algorithm and Support Vector Machine Based Smote Technique*	▪ analisis sentimen ▪ Naive Bayes dan Support Vector Machine ▪ SMOTE ▪ sentimen komentar pengguna Gojek dan *Grab* di *Google Play*	▪ analisis sentimen ▪ BackPropagation Neural Network ▪ SMOTE ▪ ulasan pengguna MyXL di Google Play Store
3	Audenza Maulana, Inayah Khasnaputri Afifah, Asghafi Mubarrak, Kiagus Rachmat Fauzan, Ardhan Dwintara Bita Parga Zen (2023)	*Comparison of Logistic Regression, MultinomialNB, SVM, and K-NN Methods on Sentiment Analysis of Gojek App Reviews on The Google Play Store*	▪ analisis sentimen ▪ Logistic Regression, MultinomialNB, SVM, and K-NN ▪ ulasan aplikasi Gojek di *Google Play Store*	▪ analisis sentimen ▪ BackPropagation Neural Network ▪ SMOTE ▪ ulasan pengguna MyXL di Google Play Store
4	Ali Mustopa, Hermanto, Anna, Eri Bayu Pratama, Ade Hendini, Deni Risdiansyah (2021)	*Analysis of User Reviews for the PeduliLindungi Application on Google Play Using the Support Vector Machine and Naive Bayes Algorithm Based on Particle Swarm Optimization*	▪ analisis sentimen ▪ Support Vector Machine dan Naive Bayes berdasarkan Particle Swarm Optimization ▪ ulasan pengguna aplikasi PeduliLindungi di *Google Play Store*	▪ analisis sentimen ▪ Backpropagation Neural Network ▪ SMOTE ▪ Ulasan pengguna MyXL di Google Play Store
5	Gilbert Darmawan, Syariful Alam, M. Imam Sulistyo (2023)	Analisis Sentimen Berdasarkan Ulasan Pengguna Aplikasi Mypertamina Pada *Google Playstore* Menggunakan Metode *Naïve Bayes*	▪ analisis sentimen ▪ Naïve Bayes ▪ Ulasan Pengguna Aplikasi Mypertamina Pada *Google Playstore*	▪ analisis sentimen ▪ Backpropagation Neural Network ▪ SMOTE ▪ ulasan pengguna MyXL di Google Play Store
6	Ernianti Hasibuan, Elmo Allistair Heriyanto (2022)	Analisis Sentimen Pada Ulasan Aplikasi *Amazon Shopping* di *Google Play Store* Menggunakan *Naive Bayes Classifier*	▪ analisis sentimen ▪ NB dengan *classifier BernoulliNB, ComplementNB, GaussianNB, dan MultinomialNB.* ▪ Ulasan Amazon Shopping	▪ analisis sentimen ▪ Backpropagation Neural Network ▪ SMOTE ▪ ulasan pengguna MyXL di Google Play Store

Tabel 2.1 : Lanjutan

No	Peneliti	Judul	Fitur	Daya Beda
7	Abitdavy Athallah Muhammad, Ermatita, Desta Sandya Prasvita (2022)	Analisis Sentimen Pengguna Aplikasi Dana Berdasarkan Ulasan Pada *Google Play* Menggunakan Metode *Support Vector Machine*	▪ analisis sentimen ▪ Support Vector Machine ▪ chi square ▪ Ulasan Pengguna Aplikasi Dana Pada *Google Playstore*	▪ analisis sentimen ▪ Backpropagation Neural Network ▪ SMOTE ▪ ulasan pengguna MyXL di Google Play Store
8	Sendi Alpin Rizaldi, Syariful Alam, Imay Kurniawan (2023)	Analisis Sentimen Pengguna Aplikasi JMO (Jamsostek *Mobile*) Pada *Google Play Store* Menggunakan Metode *Naive Bayes*	▪ analisis sentimen ▪ Naive Bayes ▪ Ulasan Pengguna Aplikasi Jamsostek *Mobile* Pada *Google Playstore* ▪ Sentimen positif 1,472 dan sentimen negatif 3,528	▪ analisis sentimen ▪ Backpropagation Neural Network ▪ SMOTE ▪ ulasan pengguna MyXL di Google Play Store

Novelty penelitian ini adalah penerapan *Backpropagation Neural Network* dan SMOTE pada analisis sentimen ulasan pengguna *MyXL di Google Play Store*, dengan mengeksplorasi pengaruh pengaturan parameter model terhadap performa pada dataset yang tidak seimbang.

K. Kerangka Konsep Penelitian

Penelitian ini menerapkan model *Backpropagation Neural Network* dan SMOTE pada analisis sentimen ulasan pengguna *MyXL di Google Play Store*. *Backpropagation* merupakan cara untuk melatih model *Artificial Neural Network* menggunakan arsitektur *Multilayer Perceptron*. Pengoptimalan model dilakukan dengan pelatihan awal menggunakan teknik *cross-validation* dan *hyperparameter tuning* secara manual. Pelatihan dilanjutkan dengan *fine-tuning* pada nilai di sekitar *learning rate* yang menghasilkan performa model terbaik menggunakan kombinasi parameter yang sesuai. Pengujian model dilakukan dengan menggunakan data *testing* dan diukur menggunakan *confusion matrix* untuk *multiclass*.

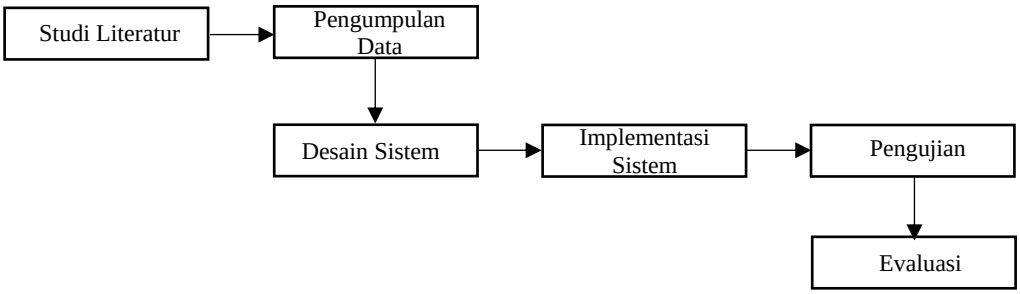
BAB III

METODOLOGI PENELITIAN

Penelitian ini merupakan analisis sentimen menggunakan model *Backpropagation Neural Network* dengan menerapkan SMOTE pada dataset Ulasan Pengguna *MyXL* di *Google Play Store* yang memiliki polaritas sentimen negatif, netral dan positif dengan jumlah tidak seimbang.

A. Desain Penelitian

Penelitian ini dilakukan dengan melalui beberapa tahapan sebagaimana ditunjukkan Gambar 3.1. Penelitian diawali dengan tahap studi literatur, dilanjutkan dengan pengumpulan data, desain sistem, implementasi sistem, pengujian, dan evaluasi.



Gambar 3.1: Desain Penelitian

B. Studi Literatur

Pengetahuan yang mendukung penelitian ini diantaranya meliputi analisis sentimen, pengolahan data teks, ekstraksi fitur, *neural network*, dan penelitian-penelitian sebelumnya. Studi literatur dilakukan pada berbagai sumber seperti jurnal penelitian, laporan penelitian, repository perpustakaan beberapa universitas,

e-book, *blog*, forum, *github.com* dan *youtube.com*.

C. Pengumpulan Data

Data yang digunakan untuk penelitian ini merupakan data sekunder. Yaitu, data yang tidak diperoleh oleh peneliti langsung dari sumbernya, tetapi merupakan data yang telah dikumpulkan oleh pihak lain yang kemudian digunakan oleh peneliti untuk dianalisis sesuai dengan keperluannya. Data tersebut adalah ulasan pengguna *MyXL* di *Google Play Store*. Dataset ini dapat diunduh di https://www.kaggle.com/datasets/dimasdiandraa/data-ulasan-terlabel?select= Ulasan+My+XL+1000+Data+Labelled.csv. Contoh ulasan pengguna *MyXL* di *Google Play Store* ditunjukkan pada Tabel 3.1.

Tabel 3.1: Contoh Ulasan Pengguna *MyXL* di *Google Play Store*

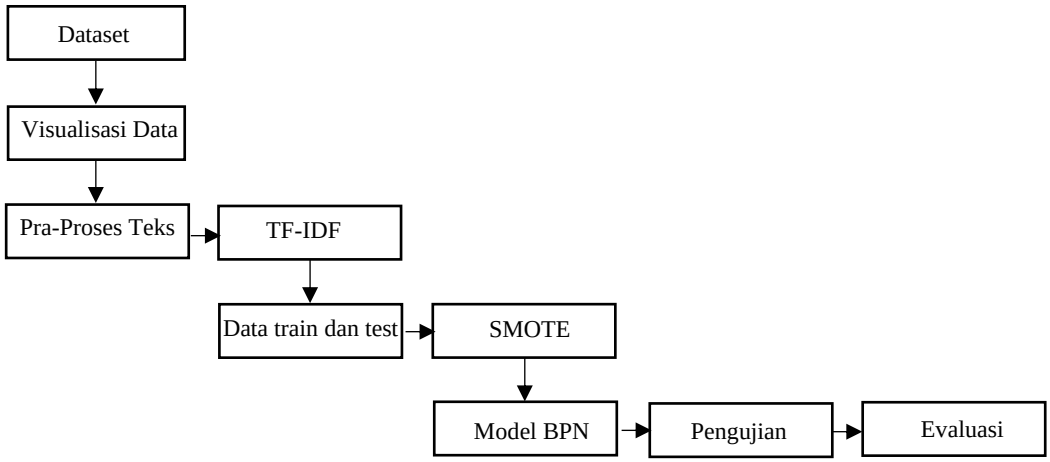
Data	Ulasan	Sentimen
Ulasan 1	********** ..... kalau bisa masa aktif kartunya di tingkatkan jadi lebih lama. terima kasih	Positif
Ulasan 2	Saya isi ulang untuk memperpanjang masa aktif. Dan sedang tidak pakai kuota. Tapi kenapa selalu berkurang pulsanya????????????	Negatif
Ulasan 3	Kembalikan pulsa saya...kuota masih banyak malah pulsa yg terpotong...	Negatif

Dataset ulasan pengguna *MyXL* di *Google Play Store* berformat .csv, terdiri dari 1,000 baris ulasan yang memiliki tiga kelas sentimen, yaitu sentimen negatif berjumlah 613 baris, sentimen netral berjumlah 226 baris dan kelas sentimen positif berjumlah 161 baris.

D. Desain Sistem

Desain sistem berguna untuk memberikan gambaran tentang proses pembangunan model *Backpropagation Neural Network* menggunakan SMOTE dan

ekstraksi fitur TF-IDF pada dataset ulasan pengguna *MyXL* di *Google Play Store* diimplementasikan dalam penelitian.



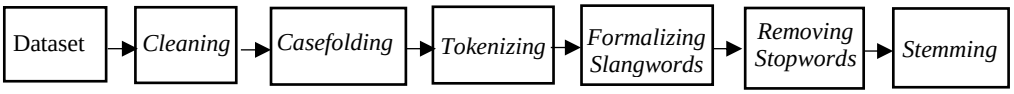
Gambar 3.2: Desain Sistem

1. Pra-Proses Teks

Preprocessing teks merupakan serangkaian teknik yang dilakukan pada data teks sehingga berubah menjadi format yang siap diolah dan dianalisis (Haikal and Hayati, 2023). Data ulasan pengguna *MyXL* pada *Google Play Store* merupakan data tidak terstruktur yang ditulis secara bebas dengan berbagai ekspresi. *Preprocessing* teks mengubah format data ini menjadi terstruktur agar dapat diolah menggunakan mesin pembelajaran. Penerapan pra-proses teks berpengaruh pada pengurangan jumlah fitur. Hal ini menjadikan algoritma semakin mudah dalam menemukan pola data saat pelatihan model berlangsung, sehingga dapat meningkatkan akurasi model. Namun, pengurangan jumlah fitur akibat penerapan pra-proses teks ini sering kali menjadi penyebab menurunnya kinerja model.

Gambar 3.3 berikut ini menunjukkan alur pra-pemrosesan teks yang merupakan serangkaian teknik yang meliputi *cleaning*, *casefolding*, *tokenizing*,

formalizing slangwords, *removing stopwords*, dan *stemming*.



Gambar 3.3: Alur Pra-pemrosesan Teks

Cleaning, teks mentah biasanya mengandung banyak *noise* yang tidak relevan dan dapat mengganggu analisis atau akurasi model. *Noise* ini sering muncul dalam bentuk *mention*, *hashtag*, angka, karakter tertentu, emoji, karakter yang berulang tiga kali atau lebih, spasi di awal dan akhir teks, spasi di awal dan akhir, terkadang teks mengandung banyak spasi kosong di antara kata-kata.

Tabel 3.2: Proses *Cleaning* pada Data Teks

Sebelum *Cleaning*	**Setelah *Cleaning***
********** kalau bisa masa aktif kartunya di tingkatkan jadi lebih lama. terima kasih	kalau bisa masa aktif kartunya di tingkatkan jadi lebih lama terima kasih
Saya isi ulang untuk memperpanjang masa aktif. Dan sedang tidak pakai kuota. Tapi kenapa selalu berkurang pulsanya????????????	Saya isi ulang untuk memperpanjang masa aktif Dan sedang tidak pakai kuota Tapi kenapa selalu berkurang pulsanya
Kembalikan pulsa saya...kuota masih banyak malah pulsa yg terpotong...	Kembalikan pulsa saya kuota masih banyak malah pulsa yg terpotong

Casefolding, menyederhanakan fitur teks dengan mengubah teks menjadi huruf kecil (*lowercase*). Proses ini bermanfaat untuk membuat indeks yang sama pada kata-kata yang awalnya berbeda penggunaan besar kecilnya huruf diubah menjadi seragam. Penyederhanaan ini dapat meningkatkan efisiensi dan akurasi pada tahap analisis atau pemrosesan teks.

Tabel 3.3 Proses *Casefolding* pada Data Teks

Sebelum *Casefolding*	Setelah *Casefolding*
kalau bisa masa aktif kartunya di tingkatkan jadi lebih lama terima kasih	kalau bisa masa aktif kartunya di tingkatkan jadi lebih lama terima kasih
Saya isi ulang untuk memperpanjang masa aktif Dan sedang tidak pakai kuota Tapi kenapa selalu berkurang pulsanya	saya isi ulang untuk memperpanjang masa aktif dan sedang tidak pakai kuota tapi kenapa selalu berkurang pulsanya
Kembalikan pulsa saya kuota masih banyak malah pulsa yg terpotong	kembalikan pulsa saya kuota masih banyak malah pulsa yg terpotong

Tokenizing, memecah teks menjadi token atau kata-kata yang terpisah membentuk satu *set array* kata. Dengan memecah teks menjadi unit-unit yang lebih kecil, analisis menjadi lebih mudah, terutama untuk algoritma yang bekerja dengan pola pada kata-kata.

Tabel 3.4: Proses *Tokenizing* pada Data Teks

Sebelum *Tokenizing*	Setelah *Tokenizing*
kalau bisa masa aktif kartunya di tingkatkan jadi lebih lama terima kasih	['kalau', 'bisa', 'masa', 'aktif', 'kartunya', 'di', 'tingkatkan', 'jadi', 'lebih', 'lama', 'terima', 'kasih']
saya isi ulang untuk memperpanjang masa aktif dan sedang tidak pakai kuota tapi kenapa selalu berkurang pulsanya	['saya', 'isi', 'ulang', 'untuk', 'memperpanjang', 'masa', 'aktif', 'dan', 'sedang', 'tidak', 'pakai', 'kuota', 'tapi', 'kenapa', 'selalu', 'berkurang', 'pulsanya']
kembalikan pulsa saya kuota masih banyak malah pulsa yg terpotong	['kembalikan', 'pulsa', 'saya', 'kuota', 'masih', 'banyak', 'malah', 'pulsa', 'yg', 'terpotong']

Formalizing slangwords, mengganti kata-kata *slang* dalam teks dengan kata-kata yang lebih formal. Kata *slang* dapat mengaburkan makna atau menimbulkan variasi berlebihan pada data. Formalisasi *slang* bertujuan untuk

membuat teks lebih seragam dan mudah dipahami oleh model NLP yang sering kali dilatih dengan bahasa formal.

Tabel 3.5: Proses *Formalizing Slangwords* pada Data Teks

Sebelum *Formalizing Slangwords*	**Setelah *Formalizing Slangwords***
['kalau', 'bisa', 'masa', 'aktif', 'kartunya', 'di', 'tingkatkan', 'jadi', 'lebih', 'lama', 'terima', 'kasih']	['kalau', 'bisa', 'masa', 'aktif', 'kartunya', 'di', 'tingkatkan', 'jadi', 'lebih', 'lama', 'terima', 'kasih']
['saya', 'isi', 'ulang', 'untuk', 'memperpanjang', 'masa', 'aktif', 'dan', 'sedang', 'tidak', 'pakai', 'kuota', 'tapi', 'kenapa', 'selalu', 'berkurang', 'pulsanya']	['saya', 'isi', 'ulang', 'untuk', 'memperpanjang', 'masa', 'aktif', 'dan', 'sedang', 'tidak', 'pakai', 'kuota', 'tapi', 'kenapa', 'selalu', 'berkurang', 'pulsanya']
['kembalikan', 'pulsa', 'saya', 'kuota', 'masih', 'banyak', 'malah', 'pulsa', 'yg', 'terpotong']	['kembalikan', 'pulsa', 'saya', 'kuota', 'masih', 'banyak', 'bahkan', 'pulsa', 'yang', 'terpotong']

Removing stopwords, menghapus kata-kata yang sering muncul dalam jumlah besar namun tidak memiliki arti penting, sehingga pemrosesan teks berfokus pada kata-kata penting.

Tabel 3.6: Proses *Removing Stopwords* pada Data Teks

Sebelum *Removing Stopwords*	**Setelah *Removing Stopwords***
['kalau', 'bisa', 'masa', 'aktif', 'kartunya', 'di', 'tingkatkan', 'jadi', 'lebih', 'lama', 'terima', 'kasih']	['aktif', 'kartunya', 'tingkatkan', 'terima', 'kasih']
['saya', 'isi', 'ulang', 'untuk', 'memperpanjang', 'masa', 'aktif', 'dan', 'sedang', 'tidak', 'pakai', 'kuota', 'tapi', 'kenapa', 'selalu', 'berkurang', 'pulsanya']	['isi', 'ulang', 'memperpanjang', 'aktif', 'pakai', 'kuota', 'berkurang', 'pulsanya']
['kembalikan', 'pulsa', 'saya', 'kuota', 'masih', 'banyak', 'bahkan', 'pulsa', 'yang', 'terpotong']	['kembalikan', 'pulsa', 'kuota', 'pulsa', 'terpotong']

Stemming, mengurangi kata ke bentuk dasar atau "akar kata" untuk mengeliminasi variasi morfologi atau mengubah teks yang memiliki imbuhan ke

bentuk kata dasarnya. Penelitian ini menggunakan *stemmer* http://sastrawi.github.io/.

Tabel 3.7: Proses *Stemming* pada Data Teks

Sebelum *Stemming*	Setelah *Stemming*
['aktif', 'kartunya', 'tingkatkan', 'terima', 'kasih']	['aktif', 'kartu', 'tingkat', 'terima', 'kasih']
['isi', 'ulang', 'memperpanjang', 'aktif', 'pakai', 'kuota', 'berkurang', 'pulsanya']	['isi', 'ulang', 'panjang', 'aktif', 'pakai', 'kuota', 'kurang', 'pulsa']
['kembalikan', 'pulsa', 'kuota', 'pulsa', 'terpotong']	['kembali', 'pulsa', 'kuota', 'pulsa', 'potong']

2. Ekstraksi Fitur TF-IDF

Setelah melalui rangkaian teknik pra-proses teks, output teks dipakai sebagai kata kunci yang akan dianalisis menggunakan pembelajaran mesin. Representasi teks akan diubah terlebih dahulu menjadi vektor numerik menggunakan teknik ekstraksi fitur yaitu pembobotan menggunakan TF-IDF. Penelitian ini menerapkan tf dan *smooth* idf yang merupakan kondisi *default* dari *library Scikitlearn*. Contoh perhitungan tf dan *smooth* idf berikut ini menggunakan tiga dokumen yang telah dibahas pada pra-proses teks.

Tabel 3.8: Contoh Token Dokumen

Dokumen	Token
dokumen 1	'aktif', 'kartu', 'tingkat', 'terima', 'kasih'
dokumen 2	'isi', 'ulang', 'panjang', 'aktif', 'pakai', 'kuota', 'kurang', 'pulsa'
dokumen 3	'kembali', 'pulsa', 'kuota', 'pulsa', 'potong'

Token-token dari ketiga dokumen tersebut adalah 'aktif', 'kartu', 'tingkat', 'terima', 'kasih', 'isi', 'ulang', 'panjang', 'pakai', 'kuota', 'kurang', 'pulsa', 'kembali',

'potong'. Pembobotan kata menggunakan TF-IDF diawali dengan menghitung frekuensi kata pada setiap dokumen (TF). Kata ‘aktif’ terdapat di dokumen 1 memiliki frekuensi 1, kata ‘aktif’ juga terdapat di dokumen 2 memiliki frekuensi 1, dan kata ‘aktif’ tidak ada di dokumen 3, maka frekuensi-nya 0. Tabel 3.9 menunjukkan frekuensi (*tf*) kata pada setiap dokumen.

Tabel 3.9: Frekuensi (TF) Setiap Kata pada Setiap Dokumen

Dokumen	aktif	kartu	tingkat	terima	kasih	isi	ulang	panjan	pakai	kuota	kurang	pulsa	kembal	potong
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	1	1	1	1	1	1	1	0	0
3	0	0	0	0	0	0	0	0	0	1	0	2	1	1

Perhitungan *idf smooth* menggunakan rumus pada Tabel 2.1. Kata ‘aktif’ terdapat di 2 dokumen dari 3 dokumen yang ada. Sehingga,

$$idf_i = \log\left(\frac{N+1}{df_i+1}\right)+1$$

$$= \log\left(\frac{3+1}{2+1}\right)+1$$

$$= \log(1{,}33)+1$$

$$= 0{,}125+1$$

$$= 1{,}125$$

Kata ‘kartu’ terdapat di 1 dokumen dari 3 dokumen yang ada. Sehingga,

$$idf_i = \log\left(\frac{N+1}{df_i+1}\right)+1$$

$$= \log\left(\frac{3+1}{1+1}\right)+1$$

$$= \log\ (2) + 1$$

$$= 0{,}3 + 1$$

$$= \ \ 1{,}3$$

Kata ‘pulsa’ terdapat di 2 dokumen dari 3 dokumen yang ada. Sehingga,

$$idf_i\ = \log\ \left(\frac{N+1}{df_i+1}\right) + 1$$

$$= \log\ \left(\frac{3+1}{2+1}\right) + 1$$

$$= \log\ (1{,}33) + 1$$

$$= 0{,}125 + 1$$

$$= 1{,}125$$

Inverse Document Frequency (*idf*) setiap kata pada setiap dokumen ditunjukkan pada Tabel 3.10.

Tabel 3.10: *Inverse Document Frequency* (IDF) Kata pada Setiap Dokumen

Dokumen	aktif	kartu	tingkat	terima	kasih	isi	ulang	panjan	pakai	kuota	kurang	pulsa	kembal	potong
1	1,1	1,3	1,3	1,3	1,3	1,3	1,3	1,3	1,3	1,1	1,3	1,1	1,3	1,3
2	1,1	1,3	1,3	1,3	1,3	1,3	1,3	1,3	1,3	1,1	1,3	1,1	1,3	1,3
3	1,1	1,3	1,3	1,3	1,3	1,3	1,3	1,3	1,3	1,1	1,3	1,1	1,3	1,3

Perhitungan bobot TF-IDF menggunakan rumus pada Tabel 2.1. Kata ‘aktif’ pada dokumen 1 memiliki tf = 1, dan idf = 1,1. Sehingga,

$$\mathrm{w}_{ij} = tf_{ij} * idf_i$$

$$= 1 * \ 1{,}1$$

$$= 1{,}1$$

Kata ‘kartu’ pada dokumen 1 memiliki tf = 1 dan idf = 1,3. Sehingga,

$$w_{ij} = tf_{ij} * idf_i$$

$$= 1 * 1{,}3$$

$$= 1{,}3$$

Kata ‘kartu’ tidak terdapat di dokumen 2, sehingga tf = 0 dan idf = 1,3. Sehingga,

$$w_{ij} = tf_{ij} * idf_i$$

$$= 0 * 1{,}3$$

$$= 0$$

Kata ‘kuota’ pada dokumen 2 memiliki tf = 1, dan idf = 1,1 Sehingga,

$$w_{ij} = tf_{ij} * idf_i$$

$$= 1 * 1{,}1$$

$$= 1{,}1$$

Kata ‘pulsa’ pada dokumen 3 memiliki tf = 2, dan idf = 1,1 Sehingga,

$$w_{ij} = tf_{ij} * idf_i$$

$$= 2 * 1{,}1$$

$$= 2{,}2$$

TF-IDF setiap kata pada setiap dokumen ditunjukkan pada Tabel 3.11.

Tabel 3.11: *Term Frequency-Inverse Document Frequency* (TF-IDF)

Dokumen	aktif	kartu	tingkat	terima	kasih	isi	ulang	panjan	pakai	kuota	kurang	pulsa	kembal	potong
1	1,1	1,3	1,3	1,3	1,3	0	0	0	0	0	0	0	0	0
2	1,1	0	0	0	0	1,3	1,3	1,3	1,3	1.1	1,3	1,1	0	0
3	0	0	0	0	0	0	0	0	0	1.1	0	2,2	1,3	1,3

3. Data *Training* dan Data *Testing*

Pembagian dataset menjadi data *training* dan data *testing* diperlukan untuk pelatihan dan pengujian model. Data *training* untuk melatih model dengan mempelajari pola-pola sentimen dalam teks dan data *testing* untuk menguji model dengan memprediksi kelas sentimen (Fachreza et al., 2023). Seribu data numerik ulasan pengguna *MyXL* pada *Google Play Store* yang dihasilkan dari ekstraksi fitur TF-IDF dibagi menjadi data *training* dan data *testing* menggunakan perbandingan 90:10. Secara random, perbandingan kelas pada data train dan data test ini sesuai dengan proposi kelas aslinya.

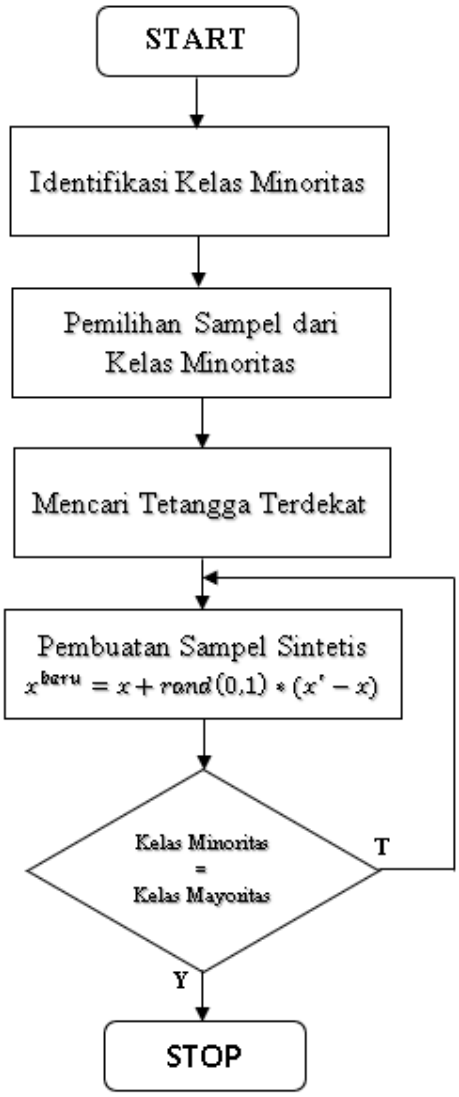
4. Alur SMOTE

SMOTE merupakan pendekatan *oversampling* yang bekerja dengan cara meningkatkan jumlah data pada kelas minoritas melalui sintesis sampel baru, sehingga jumlah data kelas minoritas tersebut sama dengan jumlah data kelas mayoritas sebelum diklasifikasikan. Gambar 3.4 menunjukkan alur *resampling* menggunakan SMOTE.

Dataset ulasan pengguna *MyXL* di *Google Play Store* memuat 1,000 baris ulasan dengan tiga kelas sentimen, yaitu kelas sentimen negatif berjumlah 613 baris ulasan, kelas sentimen netral berjumlah 226 baris ulasan dan kelas sentimen positif berjumlah 161 baris ulasan. Teknik SMOTE akan menyeimbangkan jumlah sampel di ketiga kelas sentimen tersebut, terutama untuk meningkatkan jumlah sampel pada kelas sentimen netral dan positif.

Representasi numerik hasil TF-IDF dari teks ulasan pengguna *MyXL* di *Google Play Store* akan diproses lebih lanjut menggunakan SMOTE mengikuti alur

yang ditunjukkan pada Gambar 3.4.



Gambar 3.4: Alur *Resampling* Menggunakan SMOTE

Identifikasi kelas minoritas; Kelas sentimen negatif merupakan kelas mayoritas, sementara kelas sentimen netral dan kelas sentimen positif merupakan kelas minoritas.

Pemilihan sampel dari kelas minoritas; SMOTE memilih beberapa vektor dari kelas minoritas sebagai sampel awal. Sebagai contoh sederhana, untuk ulasan

positif, SMOTE memilih sebuah vektor sampel awal untuk dijadikan basis pembuatan sampel sintetis.

Ulasan 1: “Aplikasi ini sangat membantu” [1 1,176 1 1,176 0 0]

Mencari tetangga terdekat; SMOTE menemukan beberapa tetangga terdekat, misal K=5, untuk setiap vektor sampel awal yang dipilih dari kelas minoritas. Dalam proses ini, SMOTE menghitung selisih antara vektor sampel awal dengan tetangga-tetangganya. Misalnya, vektor positif berikut ini adalah salah satu tetangga terdekat dari vektor sampel awal yang telah ditentukan sebelumnya.

Ulasan 2: “Fitur aplikasi sangat bermanfaat” [1 0 1 0 1,176 1,176]

SMOTE akan menghitung selisih kedua vektor dengan mengurangkan setiap nilai fitur dari vektor yang sesuai.

$$\Delta x = x'_{ulasan\ 2} - x_{ulasan\ 1}$$

$$= [1 \quad 0 \quad 1 \quad 0 \quad 1{,}176 \quad 1{,}176] - [1 \quad 1{,}176 \quad 1 \quad 1{,}176 \quad 0 \quad 0]$$

$$= [0 \quad -1{,}176 \quad 0 \quad -1{,}176 \quad 1{,}176 \quad 1{,}176]$$

Pembuatan sampel sintetis; SMOTE akan melakukan interpolasi dengan memilih sebuah titik di sepanjang selisih vektor antara sampel awal dan tetangga terdekat sebagai sampel sintetis, sesuai faktor pengali yang bernilai random antara 0 dan 1. Hal ini memungkinkan SMOTE untuk menghasilkan sampel sintetis dengan variasi yang mendekati makna kedua ulasan. Sampel sintetis ini bukan sekadar salinan ulasan, melainkan kombinasi dari fitur-fitur ulasan asli dan tetangganya.

Misalkan faktor pengali random antara 0 dan 1 yang digunakan adalah 0,5, maka

$$x^{baru} = x + rand(0{,}1) * (x' - x)$$

$$= x_{ulasan\ 1} + 0{,}5 * \Delta x$$

$$= [1 \quad 1{,}176 \quad 1 \quad 1{,}176 \quad 0 \quad 0] + 0{,}5 * [1 \quad 0 \quad 1 \quad 0 \quad 1{,}176 \quad 1{,}176]$$

$$= [1 \quad 1{,}176 \quad 1 \quad 1{,}176 \quad 0 \quad 0] \quad + \quad [0{,}5 \quad 0 \quad 0{,}5 \quad 0 \quad 0{,}59 \quad 0{,}59]$$

$$= [1{,}5 \; 1{,}176 \quad 1{,}5 \quad 1{,}176 \quad 0{,}59 \quad 0{,}59]$$

Jadi vektor sintetis yang tercipta adalah $[1{,}5 \; 1{,}176 \quad 1{,}5 \quad 1{,}176 \quad 0{,}59 \quad 0{,}59]$ Vektor sintetis ini mewakili ulasan baru yang berisi kata-kata yang mendekati gabungan ulasan sampel awal dan ulasan tetangganya. Misalnya,

Ulasan sintetis 1: “Aplikasi fitur sangat bermanfaat membantu” atau

Ulasan sintetis 2: “Fitur ini sangat membantu dan bermanfaat"

Dataset yang seimbang; Pembuatan sampel sintetis terus diulang untuk mendapatkan sampel sintetis baru pada setiap sampel awal di kelas minoritas positif sampai jumlah sampel pada kelas minoritas positif menjadi setara dengan jumlah sampel di kelas mayoritas negatif. Proses pembuatan sampel sintetis ini juga dilakukan pada kelas minoritas netral. Sehingga, jumlah sampel ketiga kelas sentimen negatif, netral dan positif menjadi seimbang.

5. Pembuatan Model BPN

Sebuah arsitektur *Back-Propagation Neural Network* seperti ditunjukkan pada Gambar 2.1 memiliki lapisan input dengan 3 *neuron* ($X_1 = 0.1$, $X_2 = 0.2$, $X_3 = 0.4$), satu lapisan tersembunyi dengan 3 *neuron* dan fungsi aktifasi *ReLU*, dan lapisan output dengan 3 *neuron* dan fungsi aktifasi *Softmax* ($Y_1 = 1$, $Y_2 = 0$, $Y_3 = 0$). *Learning rate* (α) = 0,01.

Proses pelatihan model *Back-Propagation Neural Network* dengan arsitektur di atas melalui tahapan-tahapan berikut ini:

a. Inisialisasi parameter/bobot secara acak menggunakan range yang kecil berdistribusi normal seragam atau standar. Misalnya, nilai bobot pada rentang [-1.1], bobot W_{i2j3} adalah 0.1 dan bias b_{j1} adalah 0.1 seperti pada matriks di bawah ini.

$$W_{ij} = \begin{bmatrix} W_{i1j1} & W_{i1j2} & W_{i1j3} \\ W_{i2j1} & W_{i2j2} & W_{i2j3} \\ W_{i3j1} & W_{i3j2} & W_{i3j3} \end{bmatrix} = \begin{bmatrix} 0.1 & 0.2 & -0.1 \\ 0.3 & 0.2 & 0.1 \\ 0.2 & -0.3 & 0.3 \end{bmatrix}$$

$$W_{jk} = \begin{bmatrix} W_{j1k1} & W_{j1k2} & W_{j1k3} \\ W_{j2k1} & W_{j2k2} & W_{j2k3} \\ W_{j3k1} & W_{j3l2} & W_{j3k3} \end{bmatrix} = \begin{bmatrix} 0.3 & 0.2 & 0.1 \\ 0.2 & -0.3 & 0.3 \\ -0.1 & -0.2 & 0.2 \end{bmatrix}$$

$$b_j = \begin{bmatrix} b_{j1} & b_{j2} & b_{j3} \end{bmatrix} = \begin{bmatrix} 0.1 & 0.2 & -0.2 \end{bmatrix}$$

$$b_k = \begin{bmatrix} b_{k1} & b_{k2} & b_{k3} \end{bmatrix} = \begin{bmatrix} 0.3 & 0.2 & 0.1 \end{bmatrix}$$

b. Feed forward

Input Layer - Hidden Layer

Setiap unit *input layer* (X_i, i = 1, 2, …., n) menerima sinyal input dan meneruskan tersebut ke seluruh unit pada *hidden layer*. Setiap unit *hidden layer* (Z_j, j = 1, 2, …, p) menjumlahkan perkalian X_i dan bobot W_{ij} ditambah dengan bias b_j seperti pada persamaan (2.10)

$$\begin{bmatrix} Z\_net_j & Z\_net_j & Z\_net_j \end{bmatrix} = \begin{bmatrix} X_{i1} & X_{i2} & X_{i3} \end{bmatrix} \times \begin{bmatrix} W_{i1j1} & W_{i1j2} & W_{i1j3} \\ W_{i2j1} & W_{i2j2} & W_{i2j3} \\ W_{i3j1} & W_{i3j2} & W_{i3j3} \end{bmatrix} + \begin{bmatrix} b_{j1} & b_{j2} & b_{j3} \end{bmatrix}$$

$$= \begin{bmatrix} 0.1 & 0.2 & 0.4 \end{bmatrix} \times \begin{bmatrix} 0.1 & 0.2 & -0.1 \\ 0.3 & 0.2 & 0.1 \\ 0.2 & -0.3 & 0.3 \end{bmatrix} + \begin{bmatrix} 0.1 & 0.2 & -0.2 \end{bmatrix}$$

$$= \begin{bmatrix} ((0.1x0.1)+0.1)+((0.2x0.3)+0.1)+((0.4x0.2)+0.1) \\ ((0.1x0.2)+0.2)+((0.2x0.2)+0.2)+((0.4x(-0.3))+0.2) \\ ((0.1x(-0.1))+(-0.2))+((0.2x0.1)+(-0.2))+((0.4x0.3)+(-0.2)) \end{bmatrix}$$

$$= \begin{bmatrix} 0.69 & -0.26 & 0.33 \end{bmatrix}$$

Output dari setiap unit *hidden layer* (Z_j) merupakan operasi *ReLU* dari input unit tersebut (Z_net_j) seperti pada persamaan (2.8).

$$\begin{aligned} Z_{j1} \quad Z_{j2} \quad Z_{j3} &= \begin{bmatrix} \max(0, Z_{in1}) & \max(0, Z_{in2}) & \max(0, Z_{in3}) \end{bmatrix} \\ &= \begin{bmatrix} \max(0, 0.69) & \max(0, -0.26) & \max(0, 0.33) \end{bmatrix} \end{aligned}$$

Hidden Layer - Output Layer

Masukan dari setiap unit *output Layer* ($\hat{y}\_net_k$) adalah jumlah dari semua perkalian output *hidden Layer* (Z_j) dan bobot (W_{jk}) ditambah bias (b_k) seperti pada persamaan (2.10)

$$\begin{bmatrix} \hat{y}\_net_{k1} & \hat{y}\_net_{k2} & \hat{y}\_net_{k3} \end{bmatrix} = \begin{bmatrix} Z_{j1} & Z_{j2} & Z_{j3} \end{bmatrix} \text{ x } \begin{bmatrix} W_{j1k1} & W_{j1k2} & W_{j1k3} \\ W_{j2k1} & W_{j2k2} & W_{j2k3} \\ W_{j3k1} & W_{j3k2} & W_{j3k3} \end{bmatrix} + \begin{bmatrix} b_{k1} & b_{k2} & b_{k3} \end{bmatrix}$$

$$= \begin{bmatrix} 0.69 & 0 & 0.33 \end{bmatrix} \text{ x } \begin{bmatrix} 0.3 & 0.2 & 0.1 \\ 0.2 & -0.3 & 0.3 \\ -0.1 & -0.2 & 0.2 \end{bmatrix} + \begin{bmatrix} 0.3 & 0.2 & 0.1 \end{bmatrix}$$

$$= \begin{bmatrix} ((0.69\text{x}0.3)+0.3)+((0\text{x}0.2)+0.3)+((0.33\text{x}(0.1))+0.3) \\ ((0.69\text{x}0.2)+0.2)+((0\text{x}(0.3))+0.2)+((0.33\text{x}(0.3))+(-0.2)) \\ ((0.69\text{x}01)+0.1)+((0\text{x}0.3)+ 0.1)+((0.33\text{x}0.2)+ 0.1) \end{bmatrix}$$

$$= \begin{bmatrix} 0.474 & 0.272 & 0.235 \end{bmatrix}$$

Keluaran dari setiap unit *output layer* ($\hat{y}_k$) adalah operasi *softmax* dari masukan unit tersebut ($\hat{y}\_net_k$) seperti pada persamaan (2.15).

$$\hat{y}_{k1} \quad \hat{y}_{k2} \quad \hat{y}_{k3} = \begin{bmatrix} e^{\hat{y}\_net_{k1}} / \sum_{k=1}^{3} e^{\text{Oink}} & e^{\hat{y}\_net_{k2}} / \sum_{k=1}^{3} e^{\text{Oink}} & e^{\hat{y}\_net_{k3}} / \sum_{k=1}^{3} e^{\text{Oink}} \end{bmatrix}$$

$$= \begin{bmatrix} \dfrac{e^{0.474}}{e^{0.474} + e^{0.272} + e^{0.235}} & \dfrac{e^{0.474}}{e^{0.474} + e^{0.272} + e^{0.235}} & \dfrac{e^{0.474}}{e^{0.474} + e^{0.272} + e^{0.235}} \end{bmatrix}$$

$$= \begin{bmatrix} 0.38395 & 0.31372 & 0.30233 \end{bmatrix}$$

c. Menghitung loss menggunakan *categorical cross-entropy* seperti pada persamaan (2.17)

$$CE = -\sum_{k=1}^{q} y_k \ . \ log(\hat{y}_k)$$

$$= -((y_1 \ . \ log(O_{out1}) + (y_2 \ . \ log(O_{out2}) + (y_1 \ . \ log(O_{out1}))$$

$$= -(((1 \ . \ log(0.38395) + (0 \ . \ log(0.31372) + (0 \ . \ log(0.30233))$$

$$= -((1 \ . \ log(0.38395) + 0 + 0)$$

$$= 0.415725998$$

d. *Backpropagation*; merupakan proses men-distribusikan kesalahan total dihitung kembali ke langkah awal (*backward*) menggunakan penurunan gradien. Metode ini diterapkan untuk memperbarui bobot, bias, sehingga kesalahan total menjadi berkurang.

Memperbarui bobot *hidden layer – output layer* (W_{jk}) dan bias *output layer* (b_k)

Turunan *error* (E) terhadap output prediksi $\hat{y}_{k1}$ seperti yang ditunjukkan persamaan (2.19) adalah

$$\frac{\partial E}{\partial \hat{y}_{k1}} = -\frac{y_{k1}}{\hat{y}_{k1}}$$

$$= -\frac{1}{0.38395}$$

$$= -2.6045$$

Turunan *error* (E) terhadap output prediksi ($\hat{y}_k$) selengkapnya ditunjukkan matriks berikut ini.

$$\begin{Bmatrix} \frac{\partial E}{\partial \hat{y}_{k1}} \\ \frac{\partial E}{\partial \hat{y}_{k2}} \\ \frac{\partial E}{\partial \hat{y}_{k3}} \end{Bmatrix} = \begin{bmatrix} -2.6045 \\ 0 \\ 0 \end{bmatrix}$$

Turunan output prediksi $\hat{y}_{k1}$ terhadap input prediksi $\hat{y}\_netk1$ yang merupakan turunan dari fungsi aktifasi *Softmax* seperti yang ditunjukkan persamaan (2.21) adalah

$$\frac{\partial \hat{y}_1}{\partial \hat{y}\_net_1} = \frac{e^{y\_net1} \,.\, (e^{y\_net2} + e^{y\_net3} + \cdots + e^{y\_netk})}{(e^{y\_net1} + e^{y\_net2} + e^{y\_net3} + \cdots + e^{y\_netk})^2}$$

$$\frac{\partial \hat{y}_1}{\partial \hat{y}\_net_1} = \frac{e^{0.474} \,.\, (e^{0.272} + e^{0.235})}{(e^{0.474} + e^{0.272} + e^{0.235})^2}$$

$$\frac{\partial \hat{y}_1}{\partial \hat{y}\_net_1} = \frac{1.60641 \,.\, (1.31259 + 1.26491)}{(1.60641 + 1.31259 + 1.26491)^2}$$

$$\frac{\partial \hat{y}_1}{\partial \hat{y}\_net_1} = \frac{1.60641 \,.\, 2.5775}{(4.1839)^2}$$

$$\frac{\partial \hat{y}_1}{\partial \hat{y}\_net_1} = \frac{4.14051}{17.505}$$

$$\frac{\partial \hat{y}_1}{\partial \hat{y}\_net_1} = \frac{1.60641 \,.\, 2.5775}{(4.1839)^2}$$

$$\frac{\partial \hat{y}_1}{\partial \hat{y}\_net_1} = 0.23653$$

Turunan output prediksi $(\hat{y}_k)$ terhadap input prediksi $(\hat{y}\_netk)$ selengkapnya ditunjukkan matriks berikut ini.

$$\begin{Bmatrix} f'(\hat{y}\_netk1) \\ f'(\hat{y}\_netk2) \\ f'(\hat{y}\_netk3) \end{Bmatrix} = \begin{Bmatrix} 0.23653 \\ 0.2153 \\ 0.21093 \end{Bmatrix}$$

Informasi *error* δ_{k1} *output layer* seperti pada persamaan (2.20) adalah:

$$\delta_{k1} = -\frac{y_{k1}}{\hat{y}_{k1}} \, \mathrm{f}'\left(\hat{y}\_net_{k1}\right)$$

$$\delta_{k1} = -2.6045 \, . \, 0.23653$$

$$\delta_{k1} = -0.6161$$

Informasi *error* δ_k output layer selengkapnya ditunjukkan matriks berikut ini.

$$\begin{Bmatrix} \delta_{k1} \\ \delta_{k2} \\ \delta_{k3} \end{Bmatrix} = \begin{Bmatrix} -0.6161 \\ 0 \\ 0 \end{Bmatrix}$$

Turunan masukan unit prediksi pada *output layer* ($\hat{y}\_net1$) terhadap bobot antara Z_{j1} dan $\hat{y}_{k1}$ (W_{j1k1}) seperti pada persamaan (2.24) adalah

$$\frac{\partial \hat{y}\_net_1}{\partial W_{j1k1}} = Z_1$$

$$\frac{\partial \hat{y}\_net_1}{\partial W_{j1k1}} = 0.69$$

Turunan masukan unit prediksi pada *output layer* ($\hat{y}\_netk$) terhadap bobot antara *hidden layer* dan *output layer* (W_{jk}) selengkapnya ditunjukkan matriks berikut ini.

$$\frac{\partial \hat{y}\_net_k}{\partial W_{jk}} = \begin{bmatrix} \frac{\partial \hat{y}\_net_{k1}}{\partial W_{j1k1}} & \frac{\partial \hat{y}\_net_{k2}}{\partial W_{j1k2}} & \frac{\partial \hat{y}\_net_{k3}}{\partial W_{j1k3}} \\ \frac{\partial \hat{y}\_net_{k1}}{\partial W_{j2k1}} & \frac{\partial \hat{y}\_net_{k2}}{\partial W_{j2k2}} & \frac{\partial \hat{y}\_net_{k3}}{\partial W_{j2k3}} \\ \frac{\partial \hat{y}\_net_{k1}}{\partial W_{j3k1}} & \frac{\partial \hat{y}\_net_{k2}}{\partial W_{j3k2}} & \frac{\partial \hat{y}\_net_{k3}}{\partial W_{j3k3}} \end{bmatrix} = \begin{bmatrix} Z_1 & Z_1 & Z_1 \\ Z_2 & Z_2 & Z_2 \\ Z_3 & Z_3 & Z_3 \end{bmatrix}$$

Memperbarui bobot W_{j1k1} dengan *learning rate* (α) = 0.1 seperti pada persamaan (2.27)

$$W_{j1k1}(\text{baru}) = W_{j1k1}(\text{lama}) - \alpha\delta_1 Z_1$$

$$= 0.3 - (0.1 * (-0.6161) * 0.69)$$

$$= 0.3 - (-0.04251)$$

$$= 0.34251$$

Bobot baru W_{jk} selengkapnya ditunjukkan matriks berikut ini.

$$W_{jk\,(baru)} = \begin{bmatrix} W_{j1k1} & W_{j1k2} & W_{j1k3} \\ W_{j2k1} & W_{j2k2} & W_{j2k3} \\ W_{j3k1} & W_{j3l2} & W_{j3k3} \end{bmatrix} = \begin{bmatrix} 0.34251 & 0.2 & 0.1 \\ 0.2 & -0.3 & 0.3 \\ -0.1 & -0.2 & 0.2 \end{bmatrix}$$

Memperbarui bias b_{k1} dengan *learning rate* (α) = 0.1 seperti pada persamaan (2.28).

$$b_{k1}(\text{baru}) = b_{k1}(\text{lama}) - \alpha\delta_{k1}$$

$$= 0.3 - (0.1 * (-0.6161))$$

$$= 0.36161)$$

Bias baru b_k selengkapnya ditunjukkan matriks berikut ini.

$$b_{k\,(baru)} = \begin{bmatrix} b_{k1} & b_{k2} & b_{k3} \end{bmatrix} = \begin{bmatrix} 0.36161 & 0.2 & 0.1 \end{bmatrix}$$

Memperbarui bobot *input layer* – *hidden layer* (W_{ij}) dan bias *hidden layer* (b_j)

Turunan *error* (E) terhadap setiap unit keluaran *hidden layer* (Z_{j1}) disimbolkan dengan $\delta_{\_net_{j1}}$ merupakan penjumlahan dari perkalian informasi *error* (δ_k) dengan bobot-bobot yang menghubungkan *hidden layer* dan *output layer* (W_{jk}) seperti pada persamaan (2.30).

$$\delta_{\_net_{j1}} = \sum_{k=1}^{m} \delta_k W_{jk}$$

$$= \delta_{k1} W_{j1k1} + \delta_{k2} W_{j1k2} + \delta_{k3} W_{j1k3}$$

$$= (-0.6161)(0.34251) + 0\ (0.2) + 0(0.1)$$

$$= -0.211001942$$

Turunan *error* (E) terhadap setiap unit keluaran *hidden layer* (Z_j) selengkapnya ditunjukkan matriks berikut ini.

$$\begin{bmatrix} \delta_{\_net_{j1}} \\ \delta_{\_net_{j2}} \\ \delta_{\_net_{j3}} \end{bmatrix} = \begin{bmatrix} -0.211001942 \\ -0.123210118 \\ 0.061605059 \end{bmatrix}$$

Turunan keluaran unit *hidden layer* Z_{j1} terhadap masukan $Z_{\_net1}$ yang merupakan fungsi aktifasi *ReLU* seperti pada persamaan (2.31) adalah

$$\text{f}'\left(z_{\_net_j}\right) \Rightarrow \quad if\ z_{\_net_j} > 0, \frac{\partial(\text{relu})}{\partial Z_{\_netj}} = 1$$

$$\text{selain itu, } \frac{\partial(\text{relu})}{\partial Z_{\_netj}} = 0$$

$$Z_{\_net1} = 0.69 \text{ , maka } \frac{\partial(\text{relu})}{\partial Z_{\_net1}} = 1$$

Turunan keluaran unit *hidden layer* Z_j terhadap masukan $Z_{\_netj}$ selengkapnya ditunjukkan matriks berikut ini.

$$\begin{bmatrix} \frac{\partial Z_{j1}}{\partial Z_{\_netj1}} \\ \frac{\partial Z_{j2}}{\partial Z_{\_netj2}} \\ \frac{\partial Z_{j3}}{\partial Z_{\_netj3}} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

Turunan masukan unit *hidden layer* ($Z_{\_net1}$) terhadap bobot antara X_1 dan

Z_1 (W_{i1j1}) seperti pada persamaan (2.24) adalah

$$\frac{\partial Z\_net1}{\partial W_{i1j1}} = X_{\mathrm{i}} = 0.1$$

Turunan masukan unit *hidden layer* (Z_net_j) terhadap bobot antara hidden layer dan output layer (W_{ij}) selengkapnya ditunjukkan matriks berikut ini.

$$\frac{\partial Z\_netj}{\partial W_{ij}} = \begin{bmatrix} \frac{\partial Z\_netj1}{\partial W_{i1j1}} & \frac{\partial Z\_netj2}{\partial W_{i1j2}} & \frac{\partial Z\_netj3}{\partial W_{i1j3}} \\ \frac{\partial Z\_netj1}{\partial W_{i2j1}} & \frac{\partial Z\_netj2}{\partial W_{i2j2}} & \frac{\partial Z\_netj3}{\partial W_{i2j3}} \\ \frac{\partial Z\_netj1}{\partial W_{i3j1}} & \frac{\partial Z\_netj2}{\partial W_{i3j2}} & \frac{\partial Z\_netj3}{\partial W_{i3j3}} \end{bmatrix} = \begin{bmatrix} X_1 & X_1 & X_1 \\ X_2 & X_2 & X_2 \\ X_3 & X_3 & X_3 \end{bmatrix}$$

Informasi *error* unit *hidden layer* (δ_{j1}) diperoleh dari perkalian δ_net_{j1} dan turunan dari fungsi aktifasi *ReLU* seperti pada persamaan (2.32).

$$\delta_{j1} = \delta\_net_{j1} . \mathrm{f}'\left(z\_net_{j1}\right)$$

$$= -0.211001942 \,.\, 1$$

$$= -0.211001942$$

Informasi *error* unit *hidden layer* (δ_j) selengkapnya ditunjukkan matriks berikut ini.

$$\begin{Bmatrix} \delta_{j1} \\ \delta_{j2} \\ \delta_{j3} \end{Bmatrix} = \begin{Bmatrix} -0.211001942 \\ 0 \\ 0.061605059 \end{Bmatrix}$$

Memperbarui bobot W_{i1j1} dengan *learning rate* (α) = 0.1 seperti pada persamaan (2.33).

$$W_{i1j1}(\mathrm{baru}) = W_{i1j1}(\mathrm{lama}) - \alpha\delta_{j1}X_{\mathrm{i1}}$$

$$= 0.1 - (0.1)(-0.211001942)(0.1)$$

$$= 0.1 - (-0.00211001942)$$

$$= 0.10211001942$$

Bobot baru W_{ij} selengkapnya ditunjukkan matriks berikut ini.

$$W_{ij\,(baru)} = \begin{bmatrix} W_{i1j1} & W_{i1j2} & W_{i1j3} \\ W_{i2j1} & W_{i2j2} & W_{i2j3} \\ W_{i3j1} & W_{i3j2} & W_{i3j3} \end{bmatrix} = \begin{bmatrix} 0.10211001942 & 0.2 & -0.10061605059 \\ 0.30422003884 & 0.2 & 0.09876789882 \\ 0.20844007768 & -0.3 & 0.29753579764 \end{bmatrix}$$

Memperbarui bias b_{j1} dengan *learning rate* (α)=0.1 seperti pada persamaan (2.34).

$$b_{j1}(\text{baru}) = b_{j1}(\text{lama}) - \alpha\delta_{j1}$$

$$= 0.1 - (0.1)(-0.211001942)$$

$$= 0.1 - (-0.0211001942)$$

$$= 0.1211001942$$

Bias baru b_j selengkapnya ditunjukkan matriks berikut ini.

$$b_{j\,(baru)} = \begin{Bmatrix} b_{j1} & b_{j2} & b_{j3} \end{Bmatrix} = \begin{Bmatrix} 0.1211001942 & -0.2 & 0.1938394941 \end{Bmatrix}$$

e. Iterasi sampai konvergen; mengulangi pelatihan dari huruf b-e sampai nilai parameter tercapai pada rata-rata *error* training rendah atau *error* mencapai batas toleransi. Pada keadaan itu iterasi akan terhenti. Iterasi juga dapat terhenti jika mencapai jumlah iterasi yang telah ditetapkan sebelumnya sekalipun belum konvergen.

E. Skenario Penelitian

Optimalisasi model dilakukan melalui pelatihan awal menggunakan teknik *cross-validation* dan *hyperparameter tuning* secara manual. *Hyperparamete*r yang sesuai dengan model *Back-Propagation Neural Network* mencakup jumlah *hidden layer*, jumlah *neuron*, dan *learning rate*. Pada *hidden layer*, digunakan konfigurasi

1 dan 2 lapisan, dengan jumlah *neuron* bervariasi antara 64, 128, dan 256, serta *learning rate* 0,1, 0,01 dan 0,001. Pelatihan awal ini menggunakan skenario seperti pada Tabel 3.12.

Tabel 3.12: Skenario Penelitian

Skena-rio	**Learning Rate**	**Jumlah Hidden layer**	**Jumlah Neuron**
1	0, 1	1	64, 128, 256
		2	64 & 64, 64 & 128, 64 & 256, 128 & 64, 128 & 128, 128 & 256, 256 & 64, 256 & 128, 256 & 256
2	0,01	1	64, 128, 256
		2	64 & 64, 64 & 128, 64 & 256, 128 & 64, 128 & 128, 128 & 256, 256 & 64, 256 & 128, 256 & 256
3	0,001	1	64, 128, 256
		2	64 & 64, 64 & 128, 64 & 256, 128 & 64, 128 & 128, 128 & 256, 256 & 64, 256 & 128, 256 & 256

Setelah pelatihan awal, proses optimalisasi model dilanjutkan dengan *fine-tuning* pada nilai di sekitar *learning rate* yang menghasilkan performa model terbaik menggunakan kombinasi parameter yang sesuai. *Fine-tuning* bertujuan untuk memperbaiki akurasi, mengurangi *error*, atau mengatasi masalah seperti *overfitting* atau *underfitting* dengan cara yang lebih mendetail daripada proses pelatihan awal.

BAB IV

PEMBAHASAN

A. Data Penelitian

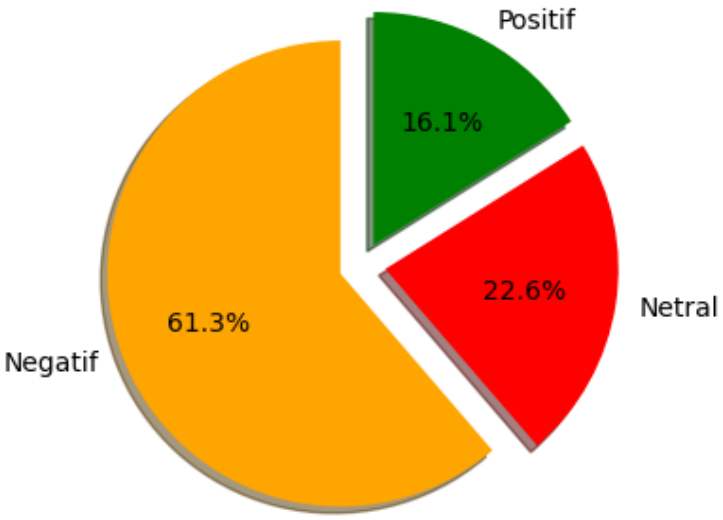
Penelitian ini menggunakan dataset yang membahas domain spesifik, yaitu ulasan pengguna *MyXL* di *Google Play Store*. Dataset memuat 1,000 baris ulasan dengan tiga kelas sentimen. Kelas sentimen negatif berjumlah 613 baris ulasan, kelas sentimen netral berjumlah 226 baris ulasan dan kelas sentimen positif berjumlah 161 baris ulasan.

Tabel 4.1 menunjukkan jumlah baris dan jumlah kelas sentimen negatif, kelas sentimen netral dan kelas sentimen positif dalam persen. Jumlah kelas sentimen negatif 61,3 %, kelas sentimen netral 22,6 % dan kelas sentimen positif 16,1 % dari keseluruhan baris ulasan dalam dataset ulasan pengguna *MyXL* di *Google Play Store*.

Tabel 4.1: Jumlah Data Setiap Sentimen

Sentimen	Jumlah (baris)	Jumlah (%)
Negatif	613	61,3
Netral	226	22,6
Positif	161	16,1
Jumlah seluruh data	1,000	100

Distribusi kelas sentimen negatif, kelas sentimen netral dan kelas sentimen positif dataset ulasan pengguna *MyXL* di *Google Play Store* secara grafis ditunjukkan pada Gambar 4.1.



Gambar 4.1: Distribusi Kelas Dataset

B. Hasil TF-IDF

Term Frequency-Inverse Document Frequency (TF-IDF) merupakan metode pembobotan teks dalam suatu dokumen sebagai langkah ekstraksi fitur yang mengubah data teks tersebut menjadi numerik agar dapat dilakukan analisis sentimen.

Rata-rata nilai TF-IDF semua fitur yang digunakan untuk model analisis sentimen ditunjukkan Tabel 4.2. Fitur "aamiin" memiliki rata-rata nilai TF-IDF sebesar 0,2507, fitur "abal" sebesar 0,8651, dan seterusnya sampai dengan fitur "yutube" memiliki rata-rata nilai TF-IDF sebesar 9,9733.

Tabel 4.2: Rata-Rata Nilai TF-IDF Semua Fitur

Index ke-	**0**	**1**	**2**	**. . .**	**1,856**	**1,857**	**1,858**
Fitur	aamiin	abal	abang	. . .	yth	yt	yutube
TF-IDF	0,2507	0,8651	0,1933	. . .	0,4507	1,1781	9,9733

C. Hasil *Train-Test Split*

Seribu data numerik ulasan pengguna *MyXL* pada *Google Play Store* yang dihasilkan dari ekstraksi fitur TF-IDF dibagi menjadi data *training* dan data *testing* menggunakan perbandingan 90:10, sebagaimana ditunjukkan pada Tabel 4.4. Secara random, perbandingan kelas pada data *training* dan data *testing* ini sesuai dengan proposi kelas aslinya.

Tabel 4.3: Jumlah Baris Data *Training* dan *Testing* Setiap Kelas

Ulasan	Jumlah Data dalam Kelas			Jumlah	%
	Kelas Negatif	Kelas Netral	Kelas Positif		
Data *training*	552	203	145	900	90
Data *testing*	61	23	16	100	10

D. Hasil SMOTE

Teknik SMOTE bertujuan untuk mengatasi ketidakseimbangan kelas pada saat proses pelatihan model berlangsung. SMOTE meningkatkan data kelas minoritas, sehingga jumlah data antar kelas menjadi seimbang dan model dapat mempelajari pola kelas minoritas dengan baik. Oleh karena itu, SMOTE hanya diterapkan pada data *training*. Data *testing* tetap dibiarkan tidak seimbang sebagai pencerminan kondisi data di dunia nyata. Distribusi data *testing* seperti ini akan menghasilkan evaluasi performa model menjadi valid, objektif, akurat dan tidak bias. Penerapan SMOTE pada seluruh dataset sebelum pembagian data *training* dan data *testing* akan mengakibatkan evaluasi performa model menjadi tidak valid dikarenakan model sudah terpengaruh oleh pola sintetis dari data *testing* selama

pelatihan. Hal ini yang disebut dengan kebocoran data. Tabel 4.4 menunjukkan jumlah sentimen data *training* sebelum dan sesudah SMOTE.

Tabel 4.4: Data *Training* Sebelum dan Sesudah SMOTE

Sentimen	Sebelum SMOTE		Sesudah SMOTE	
	Jumlah (baris)	Jumlah (%)	Jumlah (baris)	Jumlah (%)
Negatif	552	61,3	552	33,33
Netral	203	22,6	552	33,33
Positif	145	16,1	552	33,33
Jumlah seluruh data	1,000	100	1,656	100

Berdasarkan Tabel 4.4 dapat diketahui bahwa perbandingan prosentase jumlah kelas sentimen negatif, sentimen netral dan sentimen positif dengan SMOTE adalah 33,33:33,33:33,33. Perbandingan ketiga kelas sentimen ini sama dengan 1:1:1. Hal ini menunjukkan bahwa ketiga kelas telah sama jumlahnya dan seimbang.

E. Optimalisasi Model Analisis Sentimen

Klasifikasi sentimen penelitian ini menggunakan model *Backpropagation Neural Networks* pada modul *Scikitlearn* dengan *Multilayer Percepron Classifier* yang menggunakan fungsi aktifasi *default*, *ReLU* pada *hidden layer*, fungsi aktifasi *softmax* pada *output layer* dan fungsi *loss, categorical cross-entrophy*. Selain itu, model ini juga menggunakan *stochastic gradient descent (sgd)* sebagai metode untuk optimasi parameter bobot dan bias, jumlah maksimum iterasi 3,000 dan random state 0.

1. Pelatihan, Pengujian dan Evaluasi Model Sesuai Skenario

Pelatihan model diawali dengan pembagian data bersih hasil pra-

pemrosesan data dan pembobotan menjadi *data training* dan *data testing* dengan perbandingan 90:10. SMOTE diterapkan pada data *training* untuk menyeimbangkan data kelas sentimen. Optimalisasi model dilakukan dengan uji coba pengaturan *hyperparameter* meliputi jumlah *hidden layer*, jumlah *neuron* dan *learning rate*. Pada *hidden layer*, digunakan konfigurasi 1 dan 2 lapisan, dengan jumlah *neuron* bervariasi antara 64, 128, dan 256, serta *learning rate* 0,1, 0,01 dan 0,001.

Selain itu, untuk membuat akurasi model menjadi akurat dan konsisten dilakukan *cross-validation* dengan *K-fold* = 10. Jumlah *K-fold* ini menunjukkan bahwa data training hasil SMOTE dibagi menjadi 10 bagian. Masing-masing bagian secara bergiliran berperan sebagai data validasi, sementara 9 bagian yang lain menjalani pelatihan model, demikian seterusnya pelatihan dilakukan berulang kali sampai semua bagian pernah berperan sebagai data *validasi*. Akurasi pelatihan model dihitung dari rata-rata akurasi 10 kali pelatihan.

Pengujian model menggunakan data *testing* yang memiliki distribusi asli. Hal ini merupakan pencerminan dari kondisi data sebenarnya di dunia nyata yang tidak seimbang. Penggunaan data *testing* yang tidak seimbang ini akan menghasilkan evaluasi performa model yang lebih valid, objektif, akurat dan tidak bias.

Ketepatan klasifikasi dihitung menggunakan *accuracy* dan F1-*score macro average* berdasarkan *confusion matrix multiclass*. *Accuracy* menunjukkan seberapa baik model membuat prediksi yang benar. F1-*score* merupakan rata-rata harmonis dari *precision*, yang menunjukkan seberapa tepat model memprediksi kelas positif

dan *recall*, yang menunjukkan seberapa baik model mendeteksi semua kasus positif yang sebenarnya.

Tabel 4.5 menunjukkan hasil evaluasi pelatihan awal dengan beberapa pola pengaturan parameter jumlah *hidden layer*, jumlah *neuron*, dan *learning rate*. sesuai skenario penelitian.

Tabel 4.5: Hasil Evaluasi Pelatihan Awal

Jumlah Hidden Layer	**Jumlah Neuron**	**Skenario 1 Lr= 0,1**		**Skenario 2 Lr= 0,01**		**Skenario 3 Lr= 0,001**	
		Acc (%)	**F1-score (%)**	**Acc (%)**	**F1-score (%)**	**Acc (%)**	**F1-score (%)**
1	64	70,00	63.50	68,00	60,61	73,00	66.70
	128	71,00	64.86	70,00	63,06	71,00	64.07
	256	70,00	63.50	68,00	60,61	70,00	63.43
2	64,64	70,00	63.93	71,00	65,63	72,00	67.15
	64, 128	**73**,00	**68.43**	71,00	66,77	72,00	67.45
	64, 256	70,00	62.59	70,00	64,41	25,00	25.36
	128, 64	72,00	65.99	**74,00**	**69,71**	**73,00**	**68.40**
	128,128	71,00	64.09	70,00	62,87	70,00	62.87
	128, 256	69,00	62,08	70,00	63,93	72,00	66.28
	256, 64	68,00	62.22	67,00	62,37	70,00	66.80
	256, 128	70,00	63.50	69,00	61,59	67,00	59.37
	256, 256	68,00	59.67	69.00	61,04	70,00	63.20

Dari tabel di atas, secara umum, model dengan konfigurasi dua *hidden layer* tampaknya memberikan performa akurasi dan F1-*score* yang sedikit lebih baik dibandingkan satu *hidden layer*, terutama dalam konfigurasi (128, 64) dan (64,128). Konfigurasi lain dengan jumlah neuron 256 cenderung memberikan hasil akurasi dan F1-s*core* yang sedikit lebih rendah. Hal ini menunjukkan bahwa jumlah *neuron* yang besar tidak selalu mampu meningkatkan performa.

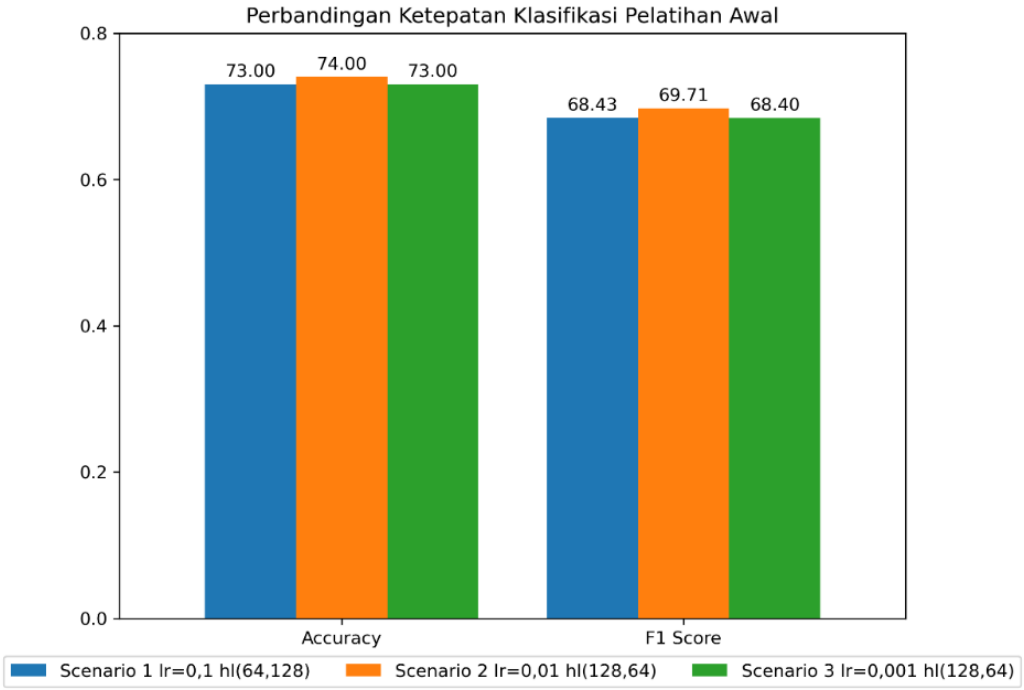
Berdasarkan Tabel 4.5, ketepatan klasifikasi tertinggi setiap skenario

penelitian pada pelatihan awal model *Backpropagation Neural Networks* ditunjukkan pada Tabel 4.6 berikut ini.

Tabel 4.6: Ketepatan Klasifikasi Tertinggi pada Pelatihan Awal

	***Accuracy* (%)**	***F1-Score* (%)**
Skenario 1	73,00	68,43
Skenario 2	74,00	69,71
Skenario 3	73,00	68,40

Ketepatan klasifikasi tiga skenario pada pelatihan awal di atas divisualisasikan pada Gambar 4.2.



Gambar 4.2: Perbandingan Ketepatan Klasifikasi pada Pelatihan Awal

Informasi dari Tabel 4.6 dan Gambar 4.2 menunjukkan bahwa Skenario 2 dengan kombinasi dua *hidden layer*, jumlah *neuron* (128, 64) dan *learning rate* 0,01

menghasilkan performa akurasi tertinggi 74% dan F1-*score* hingga 69,71% dibandingkan dengan dua skenario lainnya. Dengan demikian, Skenario 2 merupakan model *Backpropagation Neural Networks* yang memiliki performa terbaik pada pelatihan awal.

Pelatihan model *Backpropagation Neural Networks* pada Skenario 2 menggunakan *cross-validation* dengan *K-fold* = 10. *Cross-validation* dilakukan dengan tujuan untuk memperoleh akurasi model yang akurat dan konsisten. Pelatihan ini menghasilkan rata-rata *accuracy* (90.58%) dan F1-*score* (90.44%). Hasil pengukuran pelatihan Skenario 2 menggunakan *cross-validation* dengan K-*fold* = 10 ditunjukkan pada Tabel 4.7.

Tabel 4.7: Pengukuran Pelatihan Setiap *Fold* pada LR=0,01 dan HL(128,64)

***Fold*-n**	***Accuracy***	***F1-Score***
Fold-1	0.89759036	0.89469784
Fold-2	0.87349398	0.87843487
Fold-3	0.94578313	0.93968254
Fold-4	0.89156627	0.89376238
Fold-5	0.90963855	0.91098481
Fold-6	0.91566265	0.91874373
Fold-7	0.92727273	0.92669043
Fold-8	0.93939394	0.94052382
Fold-9	0.87272727	0.85714286
Fold-10	0.88484848	0.88345222
Rata-rata	0.90579774	0.90441155

Pengujian model *Backpropagation Neural Networks* dilakukan dengan menggunakan data *testing*. Hasil pengujian ini ditunjukkan dalam bentuk *confusion matrix* seperti pada Tabel 4.8.

Tabel 4.8: *Confusion Matrix* pada LR=0,01 dan HL(128,64)

Label Aktual	Label Prediksi		
	Negatif	**Netral**	**Positif**
Negatif	50	9	2
Netral	9	12	2
Positif	2	2	12

Dari Tabel 4. 8 diperoleh informasi bahwa sentimen negatif diklasifikasikan secara benar ke dalam kategori tersebut sebanyak 50 ulasan. Sedangkan 11 ulasan diklasifikasikan salah ke dalam kategori lainnya, yaitu 9 ulasan ke dalam kategori sentimen netral dan 2 ulasan dikategorikan sentimen positif. Sentimen netral diklasifikasikan secara benar ke dalam kategori tersebut sebanyak 12 ulasan. Sedangkan 11 ulasan diklasifikasikan salah ke dalam kategori lainnya, yaitu 9 ulasan ke dalam kategori sentimen negatif dan 2 ulasan dikategorikan sentimen positif. Sentimen positif diklasifikasikan secara benar ke dalam kategori tersebut sebanyak 12 ulasan. Sedangkan terdapat 4 ulasan diklasifikasikan salah ke dalam kategori lainnya, yaitu 2 ulasan ke dalam kategori sentimen negatif dan 2 ulasan dikategorikan sentimen netral.

Berdasarkan Tabel 4.8, *confusion matrix* diuraikan menjadi *True Positive* (TP), *True Negative* (TN), *False Positive* (PP), dan *False Negative* (FN), untuk masing-masing kelas sebagaimana ditunjukkan Tabel 4.9.

Tabel 4.9: TP, FP, TN, FN per Kelas pada LR=0,01 dan HL(128,64)

Kelas	TP	FP	TN	FN
Negatif	50	11	28	11
Netral	12	11	66	11
Positif	12	4	80	4

Ketepatan klasifikasi *multiclass* diukur dengan menggunakan *accuracy* dan *macro averaging* untuk *precision*, *recall* dan *f1-score*. Dengan menggunakan rumus pada Tabel 2.3, model *Backpropagation Neural Networks* dengan parameter *learning rate*=0,01, dua hidden layer dengan jumlah neuron 128 pada hidden layer pertama dan 64 pada hidden layer kedua diperoleh ketepatan klasifikasi *multiclass accuracy* (74,00%) dan *f1-score* (69,71%) seperti ditunjukkan pada Tabel 4.10.

Tabel 4.10: Ketepatan Klasifikasi pada LR=0,01 dan HL (128,64)

***Accuracy* (%)**	***F1-Score* (%)**
74,00	69,71

2. *Fine-Tuning* Nilai *Learning Rate* di Sekitar 0,01

Optimalisasi model menggunakan *fine-tuning* berguna untuk mendapatkan pengaturan parameter optimal yang memungkinkan model mencapai hasil terbaik. *Fine-tuning* dilakukan pada Skenario 2 dengan nilai *learning rate* di sekitar 0,01 menggunakan kombinasi parameter dua *hidden layer* dan jumlah neuron (128,64) yang telah menunjukkan performa model terbaik. Tabel 4.11 menunjukkan hasil evaluasi dari *fine-tuning* nilai *learning rate* di sekitar 0,01.

Tabel 4.11: *Fine-Tuning* di sekitar LR=0,01 pada HL(128,64)

Learning Rate	***Accuracy* (%)**	**F1-*score* (%)**
0,002	73,00	68,55
0,003	74,00	69,71
0,004	74,00	69,71
0,005	74,00	69,71
0,006	74,00	69,71
0,007	74,00	69,71

Tabel 4.11: Lanjutan

Learning Rate	***Accuracy* (%)**	**F1-*score* (%)**
0,008	74,00	69,71
0,009	74,00	69,71
0,01	74,00	69,71
0,011	74,00	69,71
0,012	74,00	69,71
0,013	73,00	68,4

Hasil *fine-tuning* pada Tabel 4.11 di atas menunjukkan sensitivitas model terhadap variasi nilai *learning rate* di sekitar 0,01 dengan konfigurasi *hidden layer* (128, 64). Fine-tuning diawali dari *learning rate* 0,002 sampai dengan 0,013. Karena pada *learning rate* 0,002 dan 0,013 ini, metrik evaluasi model mulai menunjukkan adanya perubahan. Dari *learning rate* 0,002 menuju 0,003, metrik evaluasi menunjukkan perubahan dari *accuracy* 73% dan F1-*score* 68,55% menuju *accuracy* 74% dan F1-*score* 69,71%. Perubahan metrik evaluasi juga terjadi dari *learning rate* 0,012 menuju 0,013. Pada *learning rate* ini, metrik evaluasi menurun dari *accuracy* 74% dan F1-*score* 69,71% menuju *accuracy* 73% dan F1-*score* 68,4%. Sementara itu, pada rentang *learning rate* 0,002 sampai dengan 0,013 ini, kestabilan performa tanpa fluktuasi terjadi pada *learning rate* antara 0,004 sampai dengan 0,011.

Dari *Learning rate* antara 0,004 sampai dengan 0,011 yang menghasilkan metrik evaluasi yang stabil ini, *fine-tuning* dilanjutkan ke nilai *learning rate* yang lebih kecil untuk memeriksa performa model lebih *detail*. *Fine-tuning* lanjutan dilakukan pada nilai tengah antara *learning rate* 0,004 sampai dengan 0,011 dan dibulatkan ke satu per-sepuluhan terdekat, yaitu 0,01.

Tabel 4.12: *Fine-Tuning* Lanjutan di sekitar LR=0,01 pada HL(128,64)

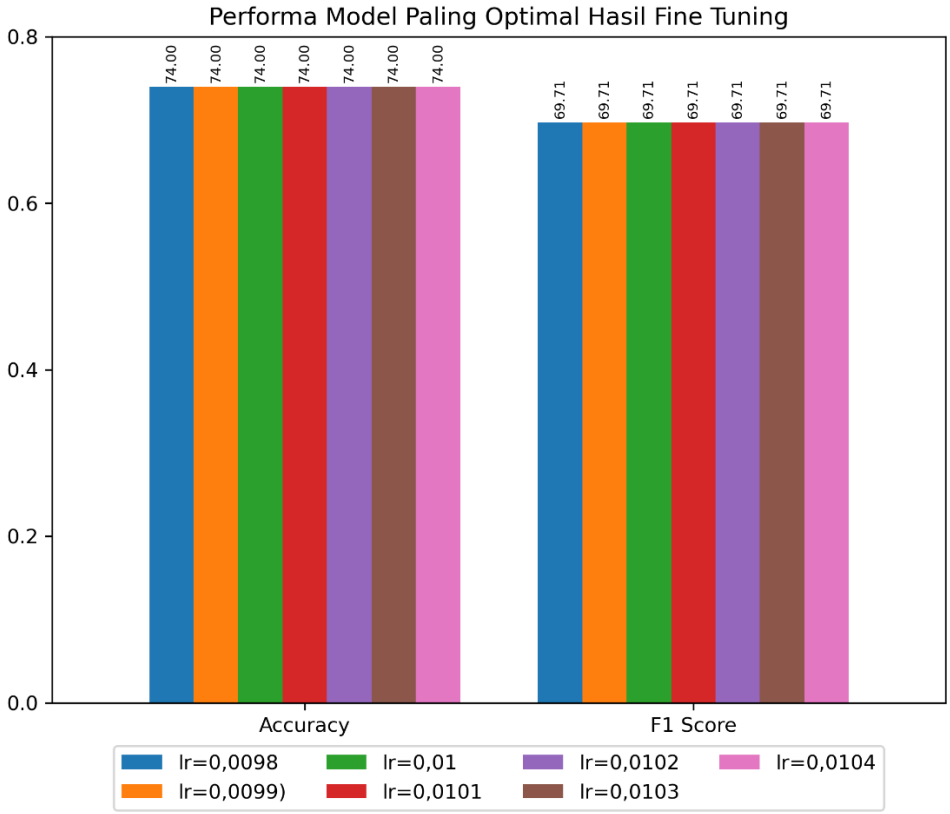
Learning Rate	*Accuracy* (%)	F1-*score* (%)
0,009	74,00	69,71
0,0091	74,00	69,71
0,0092	74,00	69,71
0,0093	74,00	69,71
0,0094	74,00	69,71
0,0095	73,00	68,40
0,0096	73,00	68,40
0,0097	74,00	69,71
0,0098	74,00	69,71
0,0099	74,00	69,71
0,01	74,00	69,71
0,0101	74,00	69,71
0,0102	74,00	69,71
0,0103	74,00	69,71
0,0104	74,00	69,71
0,0105	74,00	69,71
0,0106	73,00	69,02
0,0107	73,00	69,02
0,0108	74,00	69,71
0,0109	74,00	69,71
0,011	74,00	69,71

Hasil *Fine-tuning* lanjutan pada Tabel 4.12 di atas menunjukkan bahwa, sebagian besar nilai *learning rate* pada rentang antara 0,009 hingga 0,011 memberikan performa yang cenderung stabil, dengan akurasi 74% dan F1-*score* 69,71%. Hal ini menunjukkan bahwa model berfungsi optimal pada rentang nilai *learning rate* antara 0,009 hingga 0,011 untuk konfigurasi *hidden layer* (128, 64).

Pada nilai *learning rate* 0,0095 dan 0,0096, akurasi menurun menjadi 73% dan F1-*score* turun menjadi 68,40%. Penurunan juga terjadi pada nilai *learning rate* 0,0106 dan 0,0107, akurasi menurun menjadi 73% dan F1-*score* turun menjadi

69,02 %. Penurunan ini menunjukkan bahwa *learning rate* yang terlalu kecil atau terlalu besar di luar rentang optimal dapat berdampak pada penurunan performa.

Berdasarkan *fine-tuning* nilai *learning rate* pada rentang antara 0,009 hingga 0,011 ini, maka performa model paling optimal adalah pada nilai *learning rate* antara 0,0098 hingga 0,0104. Nilai-nilai *learning rate* ini menghasilkan akurasi dan F1-*score* terbaik tanpa fluktuasi. Performa model paling optimal ini ditunjukkan pada Gambar 4.3.



Gambar 4.3: Performa Model Paling Optimal Hasil *Fine-Tuning*

Nilai *learning rate* antara 0,0098 hingga 0,0104 menggunakan kombinasi parameter 2 *hidden layer* dan jumlah neuron (128, 64) menghasilkan *accuracy* (74,00%) dan F1-*score* (69,71%), dan merupakan performa model paling optimal.

F. Integrasi dengan Al-Qur’an

Agama Islam mengatur setiap segi kehidupan umatnya. Mengatur hubungan seorang hamba dengan Tuhannya (Allah) yang biasa disebut dengan *muamalah ma’allah* dan mengatur pula hubungan dengan sesama manusia yang biasa disebut dengan *muamalah ma’annas*.

Firman Allah dalam Al-Qur'an, Surat An-Nahl ayat 78.

وَاللّٰهُ اَخْرَجَكُمْ مِّنْۢ بُطُوْنِ اُمَّهٰتِكُمْ لَا تَعْلَمُوْنَ شَيْـًٔاۙ وَّجَعَلَ لَكُمُ السَّمْعَ وَالْاَبْصَارَ وَالْاَفْـِٕدَةَۙ لَعَلَّكُمْ تَشْكُرُوْنَ ۝

Allah mengeluarkan kamu dari perut ibumu dalam keadaan tidak mengetahui sesuatu pun dan Dia menjadikan bagi kamu pendengaan, penglihatan, dan hati nurani agar kamu bersyukur.

Menurut Terjemah Tafsir Ibnu Katsir (2024), Allah menganugerahkan pendengaran, penglihatan, hati yang terhubung dengan akal di otak, sehingga manusia mampu memahami berbagai hal serta membedakan perkara yang bermanfaat dan perkara yang *madharat*. Dengan anugerah kekuatan dan panca indra itu juga, manusia mampu berpikir, mempelajari ilmu pengetahuan dan memahami tanda-tanda kebesaran Allah SWT. Dengan daya nalarnya, manusia beriman akan *ikhlas* mengakui kebesaran-Nya dan mensyukuri pemberian-Nya yang ditunjukkan dengan tunduk patuh mengabdi beribadah, bertaqwa dengan menjalankan perintah dan menjauhi larangan-Nya.

Penelitian tentang penerapan *Backpropagation Neural Network* dan SMOTE dalam analisis sentimen ulasan pengguna *MyXL* di *Google Play Store* ini merupakan perwujudan rasa syukur atas anugerah Allah SWT. berupa kesempurnaan jiwa dan raga terutama anugerah hati, indera dan akal. Aspek ini

merupakan *muamalah ma'allah* berupa rasa syukur dengan mempelajari ilmu pengetahuan, menjalani perintah dan menjauhi larangan-Nya, ikhlas mengabdi, dan ta'at beribadah hanya kepada-Nya.

Allah SWT. memerintahkan manusia agar saling menolong antar sesama manusia dalam berbuat kebajikan dan takwa. Sebagaimana firman Allah dalam Al-Qur'an, Surat An-Maidah ayat 2.

يٰٓاَيُّهَا الَّذِيْنَ اٰمَنُوْا لَا تُحِلُّوْا شَعَاۤىِٕرَ اللّٰهِ وَلَا الشَّهْرَ الْحَرَامَ وَلَا الْهَدْيَ وَلَا الْقَلَاۤىِٕدَ وَلَآ
اٰۤمِّيْنَ الْبَيْتَ الْحَرَامَ يَبْتَغُوْنَ فَضْلًا مِّنْ رَّبِّهِمْ وَرِضْوَانًاۗ وَاِذَا حَلَلْتُمْ فَاصْطَادُوْاۗ وَلَا يَجْرِمَنَّكُمْ
شَنَاٰنُ قَوْمٍ اَنْ صَدُّوْكُمْ عَنِ الْمَسْجِدِ الْحَرَامِ اَنْ تَعْتَدُوْا وَتَعَاوَنُوْا عَلَى الْبِرِّ وَالتَّقْوٰىۖ وَلَا
تَعَاوَنُوْا عَلَى الْاِثْمِ وَالْعُدْوَانِۖ وَاتَّقُوا اللّٰهَۗ اِنَّ اللّٰهَ شَدِيْدُ الْعِقَابِ ۝

Wahai orang-orang yang beriman, janganlah kamu melanggar syiar-syiar (kesucian) Allah, jangan (melanggar kehormatan) bulan-bulan haram, jangan (mengganggu) hadyu (hewan-hewan kurban) dan qalā'id (hewan-hewan kurban yang diberi tanda), dan jangan (pula mengganggu) para pengunjung Baitulharam sedangkan mereka mencari karunia dan rida Tuhannya! Apabila kamu telah bertahalul (menyelesaikan ihram), berburulah (jika mau). Janganlah sekali-kali kebencian(-mu) kepada suatu kaum, karena mereka menghalang-halangimu dari Masjidilharam, mendorongmu berbuat melampaui batas (kepada mereka). Tolong-menolonglah kamu dalam (mengerjakan) kebajikan dan takwa, dan jangan tolong-menolong dalam berbuat dosa dan permusuhan. Bertakwalah kepada Allah, sesungguhnya Allah sangat berat siksaan-Nya.

Menurut (Shihab, 2021), saling menolong antar sesama manusia dalam berbuat kebajikan dan takwa. dan tidak saling menolong dalam berbuat dosa dan pelanggaran merupakan prinsip dasar dalam menjalin kerjasama dengan siapa pun.

Berdasarkan Surat An-Maidah ayat 2 di atas, penelitian tentang penerapan *Backpropagation Neural Network* dan SMOTE dalam analisis sentimen ulasan pengguna *MyXL* di *Google Play Store* ini dapat menjadi sarana untuk menolong dalam kebajikan, membantu dan memudahkan kehidupan manusia. Hasil penelitian

ini dapat bermanfaat bagi perusahaan PT. XL Axiata dalam meningkatkan layanan berdasarkan kebutuhan pengguna yang diungkapkan melalui sentimen ulasan mereka. Aspek ini merupakan *mu'amalah ma'annas* berupa saling menolong dalam berbuat kebajikan dan berbuat baik kepada sesama yang merupakan bentuk ketaqwaan terhadap Allah SWT.

BAB V

KESIMPULAN DAN SARAN

A. Kesimpulan

1. Penerapan model *Backpropagation Neural Network* dan SMOTE pada analisis sentimen ulasan pengguna *MyXL* di *Google Play Store* mendapatkan performa model terbaik pada nilai *learning rate* antara 0,0098 hingga 0,0104 dengan menggunakan kombinasi parameter dua *hidden layer* dengan jumlah neuron 128 pada *hidden layer* pertama dan 64 pada *hidden layer* kedua.
2. Penerapan model *Backpropagation Neural Network* dan SMOTE pada analisis sentimen ulasan pengguna *MyXL* di *Google Play Store* menggunakan kombinasi parameter *learning rate* antara 0,0098 hingga 0,0104, jumlah neuron 128 pada *hidden layer* pertama dan 64 pada *hidden layer* kedua mendapatkan ketepatan klasifikasi *macro average accuracy* (74,00%) dan F1-*score* (69,71%).

B. Saran

1. Pembaca dapat memperdalam penelitian dengan menggunakan kombinasi parameter berbeda baik jenis maupun jumlahnya untuk menemukan model *Backpropagation Neural Network* yang lebih optimal.
2. Pembaca juga dapat memperdalam penelitian dengan menggunakan metode sampling selain SMOTE.
3. Penemuan kombinasi parameter terbaik untuk mendapatkan model *Backpropagation Neural Network* yang optimal dapat dilakukan dengan memanfaatkan object *GridSearchCV* dan *RandomSearchCV* dari *ScikitLearn*.

DAFTAR PUSTAKA

Alex Yu, C.H., 2022. Data Mining and Exploration: From Traditional Statistics to Modern Data Science, 1st ed. CRC Press, New York. https://doi.org/10.1201/9781003153658

Amrullah, A.Z., Anas, A.S., Hidayat, M.A.J., 2020. Analisis Sentimen Movie Review Menggunakan Naive Bayes Classifier Dengan Seleksi Fitur Chi Square. https://doi.org/10.30812/bite.v2i1.804

Anandarajan, M., Hill, C., Nolan, T., 2019. Practical Text Analytics: Maximizing the Value of Text Data, Advances in Analytics and Data Science. Springer International Publishing, Cham. https://doi.org/10.1007/978-3-319-95663-3

Annissa, J., Ariesta, A., 2023. Analisis Semiotika Makna Nasionalisme melalui Text Mining pada Media Sosial Twitter di Kejuaraan AFF Tahun 2020. J. IPTEKKOM J. Ilmu Pengetah. Teknol. Inf. 25, 69–84. https://doi.org/10.17933/iptekkom.25.1.2023.69-84

Arsi, P., Waluyo, R., 2021. Analisis Sentimen Wacana Pemindahan Ibu Kota Indonesia Menggunakan Algoritma Support Vector Machine (SVM). J. Teknol. Inf. Dan Ilmu Komput. 8, 147. https://doi.org/10.25126/jtiik.0813944

Asrumi, A., Suharijadi, D., Setiari, A.D., Wulanda, D.P., 2023. Analisis Sentimen dan Penggalian Opini. Eureka Media Aksara, Jawa Tengah.

Audiansyah, D.D., Ratnawati, D.E., Hanggara, B.T., 2022. Analisis Sentimen Aplikasi MyXL menggunakan Metode Support Vector Machine berdasarkan Ulasan Pengguna di Google Play Store. J. Pengemb. Teknol. Inf. Dan Ilmu Komput. 6, 3987–3994.

Campesato, O., 2021. NLP fundamentals for developers. Mercury Learning and Information, Duxbury.

Chawla, N. V, Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P., 2002. SMOTE: Synthetic Minority Over-sampling Technique, Journal of Artificial Intelligence Research.

Fachreza, M.R.D., Suhartono, S., Yaqin, M.A., 2023. Klasifikasi Sentimen Masyarakat Terhadap Proses Pemindahan Ibu Kota Negara (IKN) Indonesia pada Media Sosial Twitter Menggunakan Metode Naïve Bayes. JISKA J. Inform. Sunan Kalijaga 8, 243–251. https://doi.org/10.14421/jiska.2023.8.3.243-251

Firmansyah, A.S., Aziz, A., Ahsan, M., 2023. Optimasi K-Nearest Neighbor Menggunakan Algoritma Smote Untuk Mengatasi Imbalance Class Pada Klasifikasi Analisis Sentimen. JATI J. Mhs. Tek. Inform. 7.

Galassi, A., Lippi, M., Torroni, P., 2021. Attention in Natural Language Processing. IEEE Trans. Neural Netw. Learn. Syst. 32, 4291–4308. https://doi.org/10.1109/TNNLS.2020.3019893

Gilbert, Alam, S., Sulistyo, M.I., 2023. Analisis Sentimen Berdasarkan Ulasan Pengguna Aplikasi Mypertamina Pada Google Playstore Menggunakan Metode Naïve Bayes. STORAGE J. Ilm. Tek. Dan Ilmu Komput. 2, 100–108. https://doi.org/10.55123/storage.v2i3.2333

Google Play [WWW Document], n.d. URL https://play.google/intl/id_id/comment-posting-policy/ (accessed 9.18.24).

Haikal, M., Hayati, U., 2023. Analisis Sentimen Terhadap Penggunaan Aplikasi Game Online Pubg Mobile Menggunakan Algoritma Naive Bayes. JATI J. Mhs. Tek. Inform. 7.

Hasibuan, E., Heriyanto, E.A., 2022. Analisis Sentimen Pada Ulasan Aplikasi Amazon Shopping di Google Play Store Menggunakan Naive Bayes Classifier. J. Tek. Dan Sci. 1, 13–24. https://doi.org/10.56127/jts.v1i3.434

Hermanto, Kuntoro, A.Y., Asra, T., Pratama, E.B., Effendi, L., Ocanitra, R., 2020. Gojek and Grab User Sentiment Analysis on Google Play Using Naive Bayes Algorithm and Support Vector Machine Based Smote Technique. J. Phys. Conf. Ser. 1641, 012102. https://doi.org/10.1088/1742-6596/1641/1/012102

Hidayat, E.Y., Hardiansyah, R.W., Affandy, A., 2021. Analisis Sentimen Twitter untuk Menilai Opini Terhadap Perusahaan Publik Menggunakan Algoritma Deep Neural Network. J. Nas. Teknol. Dan Sist. Inf. 7, 108–118. https://doi.org/10.25077/TEKNOSI.v7i2.2021.108-118

IPP, I.H.R.H., S. Kom., M. Kom, M. Si, D.M., S.T., 2024. KECERDASAN BUATAN. Cendikia Mulia Mandiri, Kota Batam.

Khare, N., Tomar, D.S., Ahirwal, M.K., Semwal, V.B., Soni, V. (Eds.), 2022. Machine Learning, Image Processing, Network Security and Data Sciences: 4th International Conference, MIND 2022, Virtual Event, January 19–20, 2023, Proceedings, Part II, Communications in Computer and Information Science. Springer Nature Switzerland, Cham. https://doi.org/10.1007/978-3-031-24367-7

Matondang, A.F., Dur, S., Cipta, H., 2024. Analisis Sentimen Jasa Ekspedisi Pengiriman Barang Menggunakan Metode Naive Bayes. Prox. J. Penelit. Mat. Dan Pendidik. Mat. 7, 377–386. https://doi.org/10.30605/proximal.v7i1.3653

Maulana, A., Afifah, I.K., Mubarrak, A., Fauzan, K.R., Dwintara, A., Zen, B.P., 2023. Comparison of Logistic Regression, Multinomialnb, SVM, And K-NN Methods on Sentiment Analysis of Gojek App Reviews on The Google Play Store. J. Tek. Inform. Jutif 4, 1487–1494. https://doi.org/10.52436/1.jutif.2023.4.6.863

Muhammad, A.A., Ermatita, E., Prasvita, D.S., 2022. Analisis Sentimen Pengguna Aplikasi Dana Berdasarkan Ulasan Pada Google Play Menggunakan Metode Support Vector Machine. Pros. Semin. Nas. Mhs. Bid. Ilmu Komput. Dan Apl. 3, 194–204.

Mustopa, A., Hermanto, Anna, Pratama, E.B., Hendini, A., Risdiansyah, D., 2020. Analysis of User Reviews for the PeduliLindungi Application on Google Play Using the Support Vector Machine and Naive Bayes Algorithm Based on Particle Swarm Optimization, in: 2020 Fifth International Conference on Informatics and Computing (ICIC). Presented at the 2020 Fifth International Conference on Informatics and Computing (ICIC), IEEE, Gorontalo, Indonesia, pp. 1–7. https://doi.org/10.1109/ICIC50835.2020.9288655

Noor, I.M., Turan, M., 2019. Sentiment Analysis using Twitter Dataset. IJID Int. J. Inform. Dev. 8, 84–94. https://doi.org/10.14421/ijid.2019.08206

Otter, D.W., Medina, J.R., Kalita, J.K., 2019. A Survey of the Usages of Deep Learning in Natural Language Processing.

PricewaterhouseCoopers, n.d. Voice of the Consumer Survey 2024 - Asia Pacific [WWW Document]. PwC. URL https://www.pwc.com/id/en/pwc-publications/industries-publications/ consumer-and-industrial-products-and-services/consumer-survey-2024-asia-pacific.html (accessed 10.16.24).

Purnamasari, D., Aji, A.B., 2023. Pengantar Metode Analisis Sentimen. Gunadarma, 1.

Rahayu, A.S., Fauzi, A., Rahmat, R., 2022. Komparasi Algoritma Naïve Bayes Dan Support Vector Machine (SVM) Pada Analisis Sentimen Spotify. J. Sist. Komput. Dan Inform. JSON 4, 349. https://doi.org/10.30865/json.v4i2.5398

Rizaldi, S.A.R., Alam, S., Kurniawan, I., 2023. Analisis Sentimen Pengguna Aplikasi JMO (Jamsostek Mobile) Pada Google Play Store Menggunakan Metode Naive Bayes. STORAGE J. Ilm. Tek. Dan Ilmu Komput. 2, 109–117. https://doi.org/10.55123/storage.v2i3.2334

Shihab, M.Q., 2021. Tafsir Al Mishbah, 14. Lentera Hati, Jakarta.

Text mining, 2023. Wikipedia.

Tsihrintzis, G.A., Jain, L.C. (Eds.), 2020. Machine Learning Paradigms: Advances in Deep Learning-based Technological Applications, Learning and Analytics in Intelligent Systems. Springer International Publishing, Cham. https://doi.org/10.1007/978-3-030-49724-8

Zhang, B., Dong, X., Hu, Y., Jiang, X., Li, G., 2023. Classification and prediction of spinal disease based on the SMOTE-RFE-XGBoost model. PeerJ Comput. Sci. 9, e1280. https://doi.org/10.7717/peerj-cs.1280