PENERAPAN METODE *NAÏVE BAYES* DALAM KLASIFIKASI *SOFTWARE REQUIREMENTS*

SKRIPSI

Oleh :
ATIKA SALSABILA ROSANA
NIM. 200605110153

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

PENERAPAN METODE *NAÏVE BAYES* DALAM KLASIFIKASI *SOFTWARE REQUIREMENTS*

SKRIPSI

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

Oleh :
ATIKA SALSABILA ROSANA
NIM. 200605110153

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

HALAMAN PERSETUJUAN

PENERAPAN METODE *NAÏVE BAYES* DALAM KLASIFIKASI *SOFTWARE REQUIREMENTS*

SKRIPSI

Oleh :
ATIKA SALSABILA ROSANA
NIM. 200605110153

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 02 Desember 2024

Pembimbing I,	Pembimbing II,
Dr. M. Ainul Yaqin, M.Kom	Dr. Totok Chamidy,M.Kom
NIP. 19761013 200604 1 004	NIP. 19691222 200604 1 001

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

Dr. Ir. Fachrul Kurniawan, M.MT, IPU
NIP. 19771020 200912 1 001

HALAMAN PENGESAHAN

PENERAPAN METODE *NAÏVE BAYES* DALAM KLASIFIKASI *SOFTWARE REQUIREMENTS*

SKRIPSI

Oleh :
ATIKA SALSABILA ROSANA
NIM. 200605110153

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 16 Desember 2024

Susunan Dewan Penguji

Ketua Penguji	: Dr. Zainal Abidin, M. Kom NIP. 19760613 200501 1 004	( )
Anggota Penguji I	: Syahiduz Zaman, M.Kom NIP. 19700502 200501 1 005	( )
Anggota Penguji II	: Dr. M. Ainul Yaqin, M.Kom NIP. 19761013 200604 1 004	( )
Anggota Penguji III	: Dr. Totok Chamidy,M.Kom NIP. 19691222 200604 1 001	( )

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

Dr. Ir. Fachrul Kurniawan, M.MT, IPU
NIP. 19771020 200912 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	: Atika Salsabila Rosana
NIM	: 200605110153
Fakultas / Program Studi	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: Penerapan Metode *Naïve Bayes* dalam klasifikasi *Software Requirements*

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

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 12 Desember 2024
Yang membuat pernyataan,

Atika Salsabila Rosana
NIM.200605110153

MOTTO

“Jika kamu berani memulainya, maka bertanggung jawablah untuk menyelesaikannya”

“It's okay if things don't always go as you wish, because this is your first time living and experiencing all of this. Thank you for working hard and doing your best.”

"Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya..."
-Q.S Al Baqarah: 286

HALAMAN PERSEMBAHAN

Dengan penuh rasa syukur kepada Allah SWT atas limpahan rahmat dan kemudahan-Nya, akhirnya skripsi ini dapat terselesaikan dengan baik.

Karya ini penulis persembahkan kepada:

Ibu tercinta, Latifah Hanim

Yang selalu mengalirkan kasih sayang, usaha terbaik, do'a-do'a tulus, dukungan, dan nasehat yang tiada henti.

Ayah tercinta, Edy Susilo

Yang senantiasa memberikan kekuatan, usaha terbaik, serta dukungan moral maupun materi.

Kakak dan Adik tercinta, Dinda Farah dan Dzaki Taufiqurrahman

Yang menjadi salah satu motivasi dan dorongan untuk terus maju hingga skripsi ini terselesaikan.

Segenap keluarga besar,

Yang selalu mengiringi perjalanan penulis dengan do'a dan dukungan.

Last but not least, teruntuk diri sendiri

Terima kasih untuk tidak menyerah, terima kasih sudah bertahan sejauh ini.

KATA PENGANTAR

Assalamualaikum Warahmatullahi Wabarakatuh.

Alhamdulillah, segala puji dan Syukur senantiasa penulis panjatkan pada Allah subhanahu wa ta'ala atas berkat Rahmat, serta hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi yang berjudul "Penerapan Metode *Naïve Bayes* dalam Klasifikasi *Software Requirements*". Sholawat serta salam tetap tercurahkan kepada Nabi Muhammad SAW. Dan semoga kita semua mendapat syafaatnya di hari akhir kelak, Aamiin.

Penulis mengucapkan rasa terima kasih yang begitu besar kepada seluruh pihak yang memberikan dukungan dan motivasi kepada penulis sehingga dapat menyelesaikan skripsi ini. Ucapan terima kasih ini penulis disampaikan kepada:

1. Prof. Dr. M. Zainuddin, M.A., selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Prof. Dr. Hj. Sri Harini, M.Si., selaku dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Ir. Fachrul Kurniawan, M.MT, IPU., selaku Ketua Program Studi Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Dr. M. Ainul Yaqin, M. Kom., selaku Dosen Pembimbing I dan Dr. Totok Chamidy,M.Kom selaku Dosen Pembimbing II, yang telah banyak memberikan arahan serta bimbingan, dalam menyelesaikan skripsi.
5. Dr. Zainal Abidin, M.Kom Dosen Penguji I dan Syahiduz Zaman, M.Kom selaku Dosen Penguji II, yang telah banyak memberikan uilmu, saran, dan arahan dalam menyelesaikan skripsi.

6. Segenap dosen, laboran, dan jajaran staff Program Studi Teknik Informatika yang telah memberikan ilmu, pengetahuan, dan dukungan selama penulis menjalani studi hingga selesainya skripsi ini.
7. Kedua orang tua tercinta, Ibu Latifah Hanim dan Ayah Edy Susilo yang selalu menjadi sumber kekuatan bagi penulis. Terima kasih karena selalu mengusahakan yang terbaik. Semoga Allah senantiasa memberikan kesehatan dan lindungan, sehingga dapat selalu berada disetiap perjalanan dan pencapaian penulis.
8. Kakak dan Adik tersayang Dinda Farah Rachmahwati dan Dzaki Taufiqurrahman Habibi beserta seluruh keluarga besar yang tiada henti memberikan do'a dan dukungan sehingga penulis mampu menyelesaikan skripsi ini.
9. Seluruh teman dan sahabat penulis Rosita Dewayanti, Adelia Tasya Vannesa, Helmy Dianty Putri, dan Salma Mustika Jatmiko termakasih atas segala doa, dukungan, saran, dan semangat yang diberikan selama menyusun skripsi.
10. Teman-teman INTEGER 2020 yang yang senantiasa selalu memberikan semangat dan dukungan dalam berjuang bersama dalam mengejar gelar S.Kom dan pengalaman di Universitas yang sama.
11. Seluruh pihak yang telah terlibat, baik secara langsung maupun tidak langsung dari awal perkuliahan hingga akhir penulisan skripsi ini.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih jauh dari kata sempurna. Maka dari itu penulis menerima saran, kritik dan masukan yang bersifat membangun sehingga skripsi ini dapat lebih dikembangkan. Penulis berharap semoga skripsi ini dapat memberikan manfaat untuk kedepannya.

Wassalamu'alaikum warahmatullahi wabarakatuh

Malang, 12 Desember 2024

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

ABSTRAK

Rosana, Atika Salsabila. 2024. **Penerapan Metode *Naïve Bayes* dalam Klasifikasi *Software Requirements***. Skripsi. Prodi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Dr. M. Ainul Yaqin, M.Kom (II) Dr. Totok Chamidy,M.Kom.

Kata kunci*:* Klasifikasi*,* Klasifikasi *Software Requirement, Software Requirement, Naïve Bayes*

Software Requirement adalah komponen penting dalam pengembangan perangkat lunak, mencakup fungsionalitas dan perilaku sistem yang perlu dikelola secara efektif. Tantangan dalam pengelolaan perubahan *Software Requirement*, yang dinamis dan terus berkembang, memerlukan pendekatan manajemen yang sistematis. Klasifikasi *Software Requirement* menjadi kunci dalam memastikan pengembangan perangkat lunak yang efisien dan berkualitas. Penelitian ini mengimplementasikan metode *Naïve Bayes* untuk melakukan klasifikasi persyaratan perangkat lunak. Kategori yang digunakan pada klasifikasi ini yaitu fungsional, nonfungsional, produk, proses, fungsional produk, fungsional proses, nonfungsional produk dan nonfungsional proses. Dalam pengujiannya menggunakan *K-Fold Cross Validation* dengan k-10, k-15 dan k-20. Hasil pengujian pada k-fold dengan kategori yaitu fungsional dan nonfungsional model mencapai performa terbaiknya pada k-15 dan k-20 dengan skor 85%. Kategori produk dan proses menghasilkan performa terbaik pada k-15, di mana semua metrik mencapai skor 74%. Dan pada kategori fungsional produk, fungsional proses, nonfungsional produk dan nonfungsional proses performa terbaik dihasilkan pada k-20 sebesar 67%.

ABSTRACT

Rosana, Atika Salsabila. 2024. **The Application of the Naïve Bayes Method in Software Requirements Classification.** Undergraduate Thesis. Informatics Engineering Study Program, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University Malang. Supervisor: (I) Dr. M. Ainul Yaqin, M.Kom (II) Dr. Totok Chamidy,M.Kom.

Key words: Classification, Software Requirement Classification, Software Requirement, *Naïve Bayes*

Software Requirements are a critical component in software development, encompassing the system's functionality and behavior that must be effectively managed. The challenge in managing Software Requirements, which are dynamic and constantly evolving, requires a systematic management approach. Classifying Software Requirements is key to ensuring efficient and high-quality software development. This study implements the Naïve Bayes method to classify software requirements. The categories used in this classification are functional, non-functional, product, process, functional product, functional process, non-functional product, and non-functional process. The testing process utilizes K-Fold Cross Validation with k values of 10, 15, and 20. The test results for the functional and non-functional categories showed the model achieved its best performance with k-15 and k-20, reaching a score of 85%. For the product and process categories, the best performance was achieved at k-15, with all metrics scoring 74%. Meanwhile, for the functional product, functional process, non-functional product, and non-functional process categories, the best performance was achieved at k-20, scoring 67%.

البحث مستخلص

،روسانا، أتيكا سالسابيلا. 2024. تطبيق طريقة "نايف بايز" في تصنيف متطلبات البرمجيات. أطروحة. قسم هندسة المعلوماتية الدكتور محمد عين (أنا) :كلية العلوم والتكنولوجيا، جامعة الدولة الإسلامية مولانا مالك إبراهيم مالانغ. المشرفون .الدكتور توتوك تشاميدي، ماجستير في علوم الكمبيوتر (الثاني) .اليقين، ماجستير في علوم الكمبيوتر

الكلمات المفتاحية: التصنيف، تصنيف متطلبات البرمجيات، متطلبات البرمجيات، طريقة نايف بايز

تعد متطلبات البرامج عنصرًا مهمًا في تطوير البرامج، حيث تغطي وظائف وسلوك النظام الذي يحتاج إلى إدارته بفعالية تُعد متطلبات البرمجيات مكونًا أساسيًا في تطوير البرمجيات، حيث تشمل وظائف النظام وسلوكه التي يجب إدارتها بشكل فعال لضمان تطوير برمجيات بكفاءة وجودة عالية. تواجه إدارة متطلبات البرمجيات تحديات كبيرة نظرًا لطبيعتها الديناميكية والمتطورة باستمرار، مما :لتصنيف متطلبات البرمجيات إلى عدة فئات Naïve Bayes يستدعي اتباع نهج إداري منهجي. في هذه الدراسة، تم تطبيق طريقة وظيفية، غير وظيفية، المنتج، العملية، المنتج الوظيفي، العملية الوظيفية، المنتج غير الوظيفي، العملية غير الوظيفية. تمت عملية الاختبار تبلغ 10، 15، و20، حيث أظهرت النتائج أن النموذج k بقيم K-Fold Cross Validation باستخدام التحقق المتقاطع بدرجة 85%، بينما حققت فئتا المنتج والعملية أفضل أداء عند k-20و k-15 حقق أفضل أداء لفئتي الوظيفية وغير الوظيفية عند بدرجة 74%. بالنسبة لفئات المنتج الوظيفي، العملية الوظيفية، المنتج غير الوظيفي، العملية غير الوظيفية، فقد حققت أفضل k-15 .بنسبة% 67 k-20 أداء عند

BAB I

PENDAHULUAN

1.1 Latar Belakang

Software Requirement atau persyaratan perangkat lunak adalah aspek penting yang digunakan dalam mengembangkan perangkat lunak dengan mendefinisikan fungsionalitas dan perilaku yang terdapat pada sistem perangkat lunak (Martins et al., 2019). Pengumpulan dan pengelolaan *Software Requirement* merupakan hal yang menantang. Dalam menjelajahi aspek-aspek dari *Software Requirement* diperoleh beberapa proses sistematis seperti mengumpulkan persyaratan, manajemen, manajemen perubahan, negosiasi, dan manajemen pengetahuan.

Tantangan terbesar dalam pengelolahan perangkat lunak yaitu mengelola perubahan persyaratan. Persyaratan dalam proses pengembangan akan terus berubah, sehingga sulit untuk melacak perubahan dan memastikan manajemennya tepat. Proses manajemen perubahan dan kerangka kerja diperlukan untuk mengatasi persyaratan yang bersifat dinamis dengan efektif. Persyaratan perangkat lunak adalah komponen penting dalam mengembangkan perangkat lunak dari berbagai aspek seperti memastikan pengumpulan, manajemen serta adaptasi (Abdullahi et al., 2021).

Software Requirement dikelompokkan menjadi beberapa jenis berdasarkan tujuan dan karakteristik. Salah satu jenis *Software Requirement* adalah persyaratan fungsional (FRs) dan persyaratan nonfungsional (NFRs) (Yaseen et al., 2020). Persyaratan fungsional dapat diartikan sebagai fungsionalitas dan perilaku spesifik

yang dimiliki oleh sistem perangkat lunak. Persyaratan fungsional menjelaskan perangkat lunak lakukan dan bagaimana perangkat lunak merespon input atau peristiwa yang berbeda. Persyaratan fungsional diungkapkan dalam bentuk tindakan, operasi, atau tugas tertentu yang harus dilakukan perangkat lunak (Rahimi et al., 2020).

Persyaratan nonfungsional berpusat pada atribut kualitas dan kendala sistem perangkat lunak. Persyaratan ini menentukan karakteristik yang dimiliki oleh perangkat lunak, diantaranya kinerja, keamanan, keandalan, kegunaan dan pemeliharaan. Persyaratan nonfungsional tidak berkaitan secara langsung dengan fungsi spesifik perangkat lunak namun persyaratan ini diperlukan dalam memastikan pengalaman pengguna yang berkualitas tinggi dan memuaskan (Yaseen et al., 2020).

Selain persyaratan fungsional dan persyaratan nonfungsional, terdapat beberapa persyaratan lain diantaranya persyaratan produk, persyaratan proses. Persyaratan produk merupakan kebutuhan atau kendala yang dikembangkan pada perangkat lunak (Bourque et al., 2014). Persyaratan proses merupakan kendala yang terjadi pada pengembangan perangkat lunak. Contoh dari persyaratan proses seperti perangkat lunak harus dikembangkan menggunakan proses rancangan umum pengadaan (Bourque et al., 2014).

Terdapat beberapa permasalahan yang terjadi pada penelitian yang dilaksanakan oleh (Alshazly et al., 2014) ketika berurusan dengan persyaratan perangkat lunak sehingga membutuhkan klasifikasi persyaratan yang membantu menyelesaikan permasalahan yang terjadi. Masalah-masalah yang terjadi

mencangkup pertama, ketidakkonsistenan dalam pengidentifikasian persyaratan yang dapat menyebabkan kebingungan dan kesalahan dalam memahami persyaratan yang akan memberikan damopak negatif pada proses pengembangan. Ketidakpahaman yang jelas dan konsisten Selanjutnya, penanganan persyaratan perangkat lunak yang kurang memadai dan hilang dapat menyebabkan masalah dalam pengembangan perangkat lunak.

Dengan melakukan klasifikasi persyaratan perangkat lunak, kita dapat memberikan kontribusi pada peningkatan kualitas perangkat lunak. Dengan mengkategorikan persyaratan berdasarkan karakteristik dan prioritas, pengembang dapat fokus pada aspek-aspek kritis dan memastikan bahwa perangkat lunak memenuhi atribut kualitas yang diinginkan.

Terakhir, inkonsistensi dalam proses persyaratan dapat menjadi tantangan. Proses ini dapat rentan terhadap inkonsistensi, yang dapat mempengaruhi akurasi dan keandalan klasifikasi otomatis. Oleh karena itu, mengidentifikasi dan menangani inkonsistensi ini menjadi langkah krusial untuk memastikan efektivitas proses klasifikasi.

Dengan dilakukannya klasifikasi *software requirement* dapat melakukan analisis secara rinci dan akurat terhadap kebutuhan sistem sehingga membantu dalam mengembangkan perangkat lunak secara efektif dan efisien. Klasifikasi *software requirement* dapat membantu mengidentifikasi dan memprioritaskan kebutuhan sistem sesuai dengan tujuan dari pengembangan perangkat lunak (Vineetha & Samuel, 2022).

Salah satu penelitian mengenai klasifikasi *Software Requirement* dilakukan oleh (Baker et al., 2019) dengan judul *Automatic Multi-class Non-Functional Software Require-ments Classification Using Neural Networks*. Penelitian ini menggunakan input berupa data yang digunakan berasal dari *International Requirement Engineering Conference's 2017 Data Challege*. Hasil dari *precision* berkisar antara 82% dan 94% *recall* berkisar diantara 76% dan 97% dengan *F-score* berkisar diantara 82% dan 92%.

Metode *Naïve Bayes* adalah suatu metode pengklasifikasian probabilitas yang sederhana. Penggunaan metode *Naive Bayes* digunakan dalam menghitung kumpulan probabilitas dengan menjumlahkan kombinasi nilai dan frekuensi serta kombinasi nilai pada dataset yang akan digunakan. Penggunaan metode *Naïve Bayes* pada penelitian ini dikarenakan pengklasifikasiannya memerlukan sejumlah kecil sebuah data latih dalam menentukan parameter variasi serta rata-rata dari variabel yang digunakan. Serta pada penelitian yang telah dilakukan oleh (Devita et al., 2018) menghasilkan kesimpulan dari hasil penelitian berupa metode *Naïve Bayes* lebih unggul dibandingkan dengan penggunaan metode *K-Nearest Neighbor* untuk klasifikasi 40 dokumen dengan akurasi sebesar 70% untuk *Naïve Bayes* serta 40% untuk *K-Nearest Neighbor.*

Adapun salah satu ayat yang yang berbicara tentang pentingnya verifikasi informasi sebelum menyebarkannya yaitu Surah Al-Hujurat (49:6):

يٰٓاَيُّهَا الَّذِيْنَ اٰمَنُوْٓا اِنْ جَاۤءَكُمْ فَاسِقٌۢ بِنَبَاٍ فَتَبَيَّنُوْٓا اَنْ تُصِيْبُوْا قَوْمًاۢ بِجَهَالَةٍ فَتُصْبِحُوْا عَلٰى مَا فَعَلْتُمْ نٰدِمِيْنَ

"Wahai orang-orang yang beriman! Jika seseorang yang fasik datang kepadamu membawa suatu berita, maka telitilah kebenarannya, agar kamu tidak mencelakakan suatu kaum karena kebodohan (kecerobohan), yang akhirnya kamu menyesali perbuatanmu itu." (Q.S. Al Hujurat 6).

Prinsip ini bisa dihubungkan dengan pentingnya melakukan verifikasi dan klasifikasi dokumen persyaratan perangkat lunak dengan hati-hati sebelum mengambil tindakan berdasarkan informasi tersebut. Meskipun ayat ini bukanlah tentang klasifikasi dokumen persyaratan perangkat lunak dalam konteks teknologi, prinsipnya masih bisa diterapkan.

Ketika mengekstrak ajaran atau prinsip dari Al-Quran untuk aplikasi dalam konteks tertentu, perlu diingat bahwa interpretasi dapat beragam, dan penggunaan prinsip-prinsip dilakukan secara hati-hati serta bijaksana sesuai dengan konteks yang sesuai. Interpretasi ayat-ayat Al-Quran harus dilakukan dengan hati-hati dan dalam konteks yang sesuai, dan ayat-ayat tersebut mungkin memiliki makna yang lebih mendalam dan beragam dalam konteks agama dan moral Islam. Klasifikasi dokumen persyaratan perangkat lunak adalah konsep teknis yang mungkin tidak secara langsung dibahas dalam Al-Quran, tetapi prinsip-prinsip umum kebijaksanaan dan keadilan tetap relevan dalam banyak aspek kehidupan, termasuk dalam konteks klasifikasi dokumen persyaratan perangkat lunak.

1.2 Pernyataan masalah

Berdasarkan latar belakang yang telah ditunjukkan diatas, pernyataan masalah pada penelitian ini adalah bagaimana mengatasi inkonsistensi dalam mengidentifikasi persyaratan perangkat lunak yang tidak memadai?

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai dari penelitian yang dilakukan ini yaitu mengklasifikasian persyaratan perangkat lunak (*software requirement*) ke dalam

kategori yang fungsional, nonfungsional, produk, proses, fungsional produk, fungsional proses, nonfungsional produk dan nonfungsional proses dengan menggunakan metode *Naïve Bayes*.

1.4 Batasan Masalah

Adapun batasan masalah dari penelitian yang dilakukan adalah dokumen yang digunakan dalam penelitian berdasarkan kategori-kategori fungsional, nonfungsional, produk, proses, fungsional produk, fungsional proses, nonfungsional produk dan nonfungsional proses. Dan metode yang digunakan berupa *Multinomial Naïve Bayes*

1.5 Manfaat Penelitian

Peneliti berharap penelitian yang dilakukan dapat memberikan manfaat sebagai berikut :

1. Meningkatkan efisensi proses pengembangan perangkat lunak dengan menggunakan metode *Naïve Bayes*
2. Meningkatkan kualitas perangkat lunak yang dapat membantu pengembangan fokus terhadap kategori-kategori fungsional, nonfungsional, produk, proses, fungsional produk, fungsional proses, nonfungsional produk dan nonfungsional proses

BAB II

STUDI PUSTAKA

2.1 Penelitian Terdahulu

Pada penelitian terdahulu yang dilakukan terdapat beberapa jurnal mengarah pada penelitian yang akan dilakukan. Beberapa penelitian diantaranya penelitian dilakukan oleh (Tiun et al., 2020) mengenai mengklasifikasikan kebutuhan fungsional, nonfungsional, produk dan proses pada *software requirement* dengan membandikan 2 metode yaitu *Word2vec* dan *FastText*. Penelitian dilakukan menggunakan data berupa *Requirement Engineering '17 Data Challenge Area*. Hasil dari perbandingan yaitu *FastText* lebih baik dibandingkan dengan *Word2vec.*

Penelitian lain dilakukan menggunakan metode *Naïve Bayes* dilakukan oleh (Hartijo et al., 2018) yang penelitiannya digunakan untuk mengklasifikasikan dokumen memuat serangan jaringan. Tujuan penggunaan metode *Multinomial Naïve Bayes* (MNB) dalam penelitian yaitu dapat mengatur sejumlah besar informasi. Selain penggunaan *Multinomial Naïve Bayes* (MNB), peneliti menggunakan *Term Frequency-Inverse Document Frequency* (TF-IDF) yang dapat membantu menghitung term yang ada pada setiap dokumennya. Peneliti menggunakan data berjumlah 1000 jurnal dengan berisi "*Network Attacks*" dari halaman website ieeexplore.ieee.org www.sciencedirect.com tahun 2014-2021. Hasil akurasi yang di dapat sebesar 76.00%. Hasil *recall*, *precision,* dan *F1-score* mendapatkan nilai *precesion* sebesar 0,77, nilai *recall* dan *F1-Score* sebesar 0,76.

Penelitian mengenai klasifikasi *Software Requirement* berdasarkan pemodelan topik dilakukan oleh (Baker et al., 2019). Penelitian ini menggunakan

input berupa data yang digunakan berasal dari *International Requirement Engineering Conference's 2017 Data Challege*. Di dalam penelitian ini menggunakan dua jenis tipe dari model *Neural Network* yaitu *Artificial Neural Network* (ANN) dan *Convolutional Neural Network* (CNN) untuk mengklasifikasikan nonfungsional perangkat lunak dengan menggunakan 5 karegori yaitu pemeliharaan, pengoperasian, kinerja, keamanan, kegunaan. Dihasilkan *precision* berkisar antara 82% dan 94% *recall* berkisar diantara 76% dan 97% dengan *F-score* berkisar diantara 82% dan 92%.

Dalam penelitian yang sudah dilakukan oleh (Devita et al., 2018) menggunakan metode *Naive Bayes* serta *K-Nearest Neighbor* sebagai metode untuk melakukan klasifikasi. Peneliti memilih metode *Naïve Bayes* karena menghasilkan akurasi lebih baik dengan hanya menggunakan data latih yang sedikit. Sedangkan pada metode *K-Nearest Neighbor*, peneliti memilih dikarenakan metode tersebut mampu mengatasi data yang mengganggu. Data yang digunakan peneliti sebanyak 40 jurnal, macam-macam jurnal di antaranya jurnal Pendidikan Bisnis dan Manajemen, Pendidikan Ekonomi, jurnal Ekonomi Bisnis, dam Akuntansi Aktual. Metode *Naive Bayes* menghasilkan tingkat akurasi 70%, sedangkan metode *K-Nearest Neighbor* menghasilkan tingkat akurasi yaitu 40%.

Penelitian lain yang memiliki judul klasifikasi dokumen dengan metode *Naive Bayes* terhadap putusan kasasi pengadilan tentang merk dilakukan oleh(Sulastri & Zuliardo, 2020). Klasifikasi yang dilakukan pada penelitian menggunakan Dokumen mengenai merk yang diambil dari https://putusan.mahkamahagung.go.id/pengadilan/mahkamahagung/direktori/perd

ata-khusus/Merk pada dokumen putusan pengadilan sebanyak 50 data. Peneliti mendapatkan kesimpulan bahwa didapatkan hasil akurasi sebesar 80% pada kata sebanyak 5 yang banyak muncul pada analisis teks diantaranya pemohon, pengadilan, kasasi, permohonan, dan peninjauan.

Tabel 2. 1 Penelitian Terdahulu yang telah Dilakukan

No	Penulis, tahun, judul	*Input*	Metode	*Output*	Represen-tasi Perantara	Keterbatasan Studi
1.	(Tiun et al., 2020), "*Classifica-tion of functional and non-functional requirement in software requirement using Word2vec and fastText"*	Dataset pada penelitian ini adalah *Requirement Engineering '17 Data Challenge Area*	*Word2ve c and fast Text*	Hasil kinerja *Doc2vec* tidak lebih baik daripada *fastText.* Penggunaan *Doc2vec* terlihat dengan skor F1 terendah yang hanya 73,47% *FastText* dianggap sebagai model dengan hasil terbaik sebesar 92,8%	*linear regression, Naïve Bayes, Support Vector Machine, Convolu-tional Neural Network*	Dataset yang digunakan tidak terlalu banyak
2.	(Hartijo et al., 2018), "Klasifikasi Dokumen berkonten Serangan jaringan Menggunа-kan *Multinomial Naive Bayes"*	Data yang digunakan memiliki jumlah 1000 jurnal dengan memuat "*Network Attacks*" dari halaman website ieeexplore. ieee.org www.science -direct.com sejak tahun 2014-2021	*Naïve Bayes*	Hasil akurasi sebesar 76.00%. Hasil *recall*, *precision*, dan *F1-score* mendapatka n nilai *Precesion* sebesar 0,77, nilai *Recall* dan nilai *F1 Score* sebesar 0,76	*Term Frequncy – Inverse Document Frequency* (TF-IDF) dan *Text Mining*	Jurnal ini tidak menyebutkan jenis serangan jaringan apa yang diklasifikasi-kan dalam penelitian.

Lanjutan Tabel Penelitian Terdahulu yang telah Dilakukan

No	Penulis, tahun, judul	*Input*	Metode	*Output*	Represen-tasi Perantara	Keterbatasan Studi
3.	(Baker et al., 2019), "*Automatic Multi-class Non-Functional Software Require-ments Classificatio n Using Neural Networks"*	Data yang digunakan berasal dari *International Requirement Engineering Conference's* 2017 *Data Challege*	*Neural Network s*	Pada penelitian ini yaitu *precision* berkisar antara 82% dan 94% *recall* berkisar diantara 76% dan 97% dengan an *F-score* berkisar diantara 82% dan 92%.		Dataset yang digunakan tidak menjelaskan berapa data yang akan digunakan.
4.	(Devita et al., 2018), "Perbanding an Kinerja Metode *Naive Bayes* dan *K-Nearest Neighbor* untuk Klasifikasi Artikel Berbahasa Indonesia"	Menggunaka n data sebanyak 40 jurnal di antaranya adalah; Pendidikan Bisnis dan Manajemen, jurnal Ekonomi Bisnis, Akuntansi Aktual, dan jurnal Pendidikan Ekonomi.	*Naive Bayes* dan *K-Nearest Neigh-bor*	Memiliki tingkat akurasi sebesar 70% pada *Naïve Bayes*, sedangkan memiliki tingkat akurasi yaitu 40% pada *K-Nearest Neighbor*.	*Term Frequncy – Inverse Document Frequency* (TF-IDF)	- Dataset yang digunakan kurang banyak sehingga diharapkan menambah-kan data pada penlitian selanjutnya. - Tahapan Preprosesing yang dilakukan tidak lengkap
5.	(Sulastri & Zuliardo, 2020), "Klasifikasi Dokumen dengan Metode *Naive Bayes* terhadap Putusan Kasasi Pengadilan tentang Merk"	Dokumen mengenai merk yang diambil di halaman https://putusa. mahkamah agung.go.id/ pengadilan/ mahkamah-agung/ direktori/ perdata-khusus/Merk pada Putusan Pengadilan.	*Naive Bayes*	Menggunaka-kan 4 data uji dan 10 data latih menghasil-kan akurasi yaitu 80%.	*Document Term Matrix* (DTM)	Dataset yang digunakan tidak konsisten dan tidak jelas jumlahnya.

Dari penelitian terdahulu yang dijabarkan di atas *novelty* dari penelitian ini adalah pengenalan kategori baru, yaitu proses dan produk, yang melengkapi kategori fungsional dan non-fungsional yang selama ini digunakan dalam penelitian sebelumnya. Kategori ini memberikan pendekatan yang lebih menyeluruh dalam menganalisis *software requirement*, sehingga menghasilkan wawasan yang lebih komprehensif mengenai pengklasifikasian *software requirement*.

2.2 *Software Requirement*

Software Requirement dapat dideskripsikan sebagai fungsi, kinerja dan batasan yang memenuhi sistem perangkat lunak yang akan digunakan. *Software Requirement* menggambarkan kebutuhan yang dinyatakan pengguna serta dirumuskan sesuai dengan standar kualitas perangkat lunak dalam bentuk kalimat (Martins et al., 2019). *Software Requirement* menjelaskan mengenai layanan, batasan, dan fitur yang digunakan oleh sistem dapat memperesentasikan pengetahuan yang diperlukan dalam pengembangannya.

Software Requirement harus terukur, spesifik, konsisten dan dapat diverifikasi. Tujuan utama adalah menyediakan penduan dengan jelas pagi para pengembang mengenai hal yang akan dibangun dan hasil yang akan dievaluasi.

Software Requirement memiliki beberapa jenis kebutuhan dengan berbagai fokus yang berbeda. Kebutuhan fungsional menerangkan mengenai apa yang harus dilakukan oleh sistem, seperti fitur-fitur yang dibutuhkan serta operasi-operasi yang mendukung. Di sisi lain, kebutuhan nonfungsional mencangkup atribut-atribut sistem contohnya kinerja, keandalan serta keamanan.

Selain itu, terdapat kebutuhan berorientasi produk yang mengutamakan fitur dan fungsionalitas yang dimiliki perangkat lunak mencangkup deskripsi menganai yang dilakukan sistem dan fitur spesifik yang ada. Selanjutnya, kebutuhan berorientasi proses yang mengutamakan pada langkah-langkah dan prosedur yang diikuti dalam mengembangkan perangkat lunak seperti memuat metode pengembangan, teknik, serta kebutuhan terkait manajemen proyek secara keseluruhan.

Kemudian, kebutuhan sisten yang memiliki kaitan dengan sistem keseluruhan perangkat lunak yang akan diimplementasikan meliputi penggabungan dengan sistem, perangkat keras dan kebutuhan infrastruktur. Terakhir adalah kebutuhan perangkat lunak yang ditunjukkan pada kebutuhan spesifik yang dipenuhi perangkat lunak dalam pengembangan yang mencangkup aspek-aspek seperti kinerja, keamanan, keandalan, dan kemudahan penggunaan perangkat lunak.

2.3 Klasifikasi *Software Requirement*

Klasifikasi *Software Requirement* membantu dalam memahami dan mengorganisir berbagai jenis kebutuhan yang harus dipenuhi selama pengembangan sebuah produk perangkat lunak. Klasifikasi kebutuhan perangkat lunak memainkan peran penting dalam pengelolaan kebutuhan perangkat lunak. Tujuannya adalah untuk mengklasifikasikan persyaratan berdasarkan tingkat signifikansi, kompleksitas, dan saling ketergantungan.

Kebutuhan perangkat lunak dapat dibagi menjadi beberapa tingkatan diantaranya tingkat rendah, tingkat menengah dan tingkat tinggi. Pada tingkat tinggi, kebutuhan perangkat lunak dianggap penting karena memiliki dampak pada

kinerja sistem. Pada tingkat menengah, kebutuhan perangkat lunak dianggap tidak sepenting tingkat tinggi namun, masih memiliki pengaruh yang cukup besar pada perilaku sistem dan kinerjanya. Dan tingkat rendah, kebutuhan perangkat lunak dianggap tidak terlalu penting karena memiliki dampak signifikan pada sistem dan kinerjanya (Nita Yunitasari et al., 2022). Klasifikasi *Software Requirement* dapat dilakukan dengan menggunakan kategori-kategori seperti berikut :

1. Kebutuhan fungsional

   Kebutuhan fungsional adalah sebuah kebutuhan memiliki hubungan dengan tindakan yang sistem inginkan(Serafintino & Susilowati, 2022). Kebutuhan fungsional mengutamakan fitus dan fungsi yang ada di dalam perangkat lunak.

2. Kebutuhan nonfungsional

   Kebutuhan nonfungsional adalah kebutuhan tidak memiliki hubungan dengan tindakan dengan contoh keamanan, kinerja, kecepatan dan kegunaan (Serafintino & Susilowati, 2022). Kebutuhan nonfungsional merujuk pada atribut-atribut sistem dengan contoh keamanan, kinerja, dan kendala.

3. Kebutuhan produk

   Kebutuhan produk merupakan kebutuhan yang memiliki hubungan dengan hasil akhir yang sesuai keinginan seperti penampilan, fitur serta, kualitas. Kebutuhan produk memiliki fitur-fitur yang diinginkan dalam produk perangkat lunak.

4. Kebutuhan proses

Kebutuhan proses merupakan kebutuhan yang memiliki hubungan dengan sistem beroperasi. Kebutuhan proses memiliki kebutuhan yang memiliki kaitan dengan proses pengujian, pengelolaan perangkat lunak serta pengembangan.

2.4 *Naïve Bayes*

Pendekatan statistik yang dikenal sebagai *Naïve Bayes*, yang berakar dalam ranah probabilitas, awalnya ditemukan oleh ilmuwan terkemuka asal Inggris, Thomas Bayes. Tujuan utama dari pendekatan ini adalah memprediksi probabilitas pada masa depan dengan mengambil wawasan di pengalaman masa lampau (Ramadandi & Jahring, 2020). Metodologi klasifikasi *Naïve Bayes* dibangun di atas prinsip-prinsip dasar probabilitas dan Teorema Bayes. Metode ini berfungsi berdasarkan asumsi bahwa setiap variabel, yang ditunjukkan sebagai X, entah itu mandiri atau tidak memiliki korelasi dengan variabel lainnya. Karakteristik unik dari metodologi ini, yang biasa disebut sebagai "Naïve," berasal dari keyakinan bahwa setiap variabel input memiliki pengaruh individual terhadap variabel target, tanpa mempertimbangkan interaksi potensial di antara variabel input (Buchori et al., 2022).

Keuntungan dalam menggunakan *Naïve Bayes* yaitu untuk melakukan klasifikasi memerlukan data latih yang terbatas (kecil) dalam menentukan varians dari variabel dan parameter mean (Devita et al., 2018). Selain itu, keuntungan menggunakan *Naïve Bayes* yaitu sederhana dan mudah dalam pengimplementasiannya. Teorema Bayes merupakan rumus dalam teori

probabililas yang digunakan untuk menghitung probabilitas peristiwa tersebut didasarkan pada pemahaman sebelumnya mengenai situasi yang berkaitan dengan kejadian tersebut. Teorema bayes dituliskan seperti berikut :

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)} \quad (2.1)$$

Dimana :
$P(A|B)$: Kemungkinan terjadinya *A* dengan syarat *B* sudah terjadi.
$P(B|A)$: Kemungkinan kejadian *B* terjadi jika *A* telah terjadi
$P(A)$: Kemungkinan awal (atau **prior**) dari kejadian *A* tanpa mempertimbangkan informasi *B*
$P(B)$: Kemungkinan awal (**marginal**) dari kejadian *B*, diperoleh dengan mempertimbangkan semua kemungkinan yang menyebabkan *B*.

Dalam hal klasifikasi, *Naïve Bayes* dapat digunakan dalam mengelompokkan kelas suatu wujud berdasarkan kategori-kategorinya.

BAB III

METODOLOGI PENELITIAN

Penelitian yang dilakukan ini memiliki beberapa tahapan yang dapat dilakukan secara sistematis dan terstruktur sehingga dapat memudahkan penulis dalam memenuhi hasil yang sesuai. Dalam memenuhi tujuan dari penelitian dibutuhkan suatu rancangan atau desain sistem yang sesuai dengan kondisi.

3.1 Data

Tahapan pertama dilakukan pada penelitian yaitu mengumpulkan beberapa data berupa *software requirement dataset.* Penelitian ini menggunakan *dataset software requirements* yang terdiri dari 100 data, diambil dari platform Kaggle dalam format file Comma-Separated Values (CSV). Pengambilan data sebanyak 100 data dikarenakan di Kaggle yang dikategorikan sebagai fungsional dan nonfungsional hanya 100 data. Data tersebut dianalisis dan dikategorikan untuk memahami karakteristik kebutuhan perangkat lunak secara lebih mendalam. Pengategorian dilakukan oleh anotator yang memahami *Software Requirements*. Proses klasifikasi dilakukan dalam beberapa tahap, yaitu berdasarkan sifat kebutuhan, tujuan kebutuhan, serta kombinasi keduanya.

Pada tahap pertama, data diklasifikasikan menjadi dua kategori utama berdasarkan sifatnya, yaitu fungsional dan nonfungsional. Kebutuhan fungsional mencakup fitur-fitur spesifik yang harus disediakan oleh perangkat lunak, seperti fungsi pencarian atau validasi data. Sementara itu, kebutuhan nonfungsional berfokus pada atribut kualitas perangkat lunak, seperti kecepatan respon,

keamanan, atau keandalan sistem. Setiap data dalam dataset dikelompokkan ke salah satu dari dua kategori ini. Kategori fungsional pada tipe kategori disingkat menjadi FR dan nonfungsional disingkat NFR. Contoh data *software requirement* dengan kategori fungsional dan nonfungsional sebagai berikut

Tabel 3. 1 Contoh Data *Software Requirement* kategori fungsional dan nonfungsional

Type	Requirement
NFR	Restaurant Owner Create Account Security GIST: The security of creating account for restaurant owners of the system. SCALE: If a restaurant owner wants to create an account and the desired user name is occupied the restaurant owner should be asked to choose a different user name. METER: Measurements obtained on 1000 hours of usage during testing. MUST: 100% of the time.
NFR	System Availability GIST: The availability of the system when it is used. SCALE: The average system availability (not considering network failing). METER: Measurements obtained from 1000 hours of usage during testing. MUST: More than 98% of the time. PLAN: More than 99% of the time. WISH: 100% of the time.
FR	Mobile application - Search DESC: Given that a user is logged in to the mobile application then the first page that is shown should be the search page. The user should be able to search for a restaurant according to several search options. The search options are Price Destination Restaurant type and Specific dish. There should also be a free text search option. A user should be able to select multiple search options in one search.
FR	The client will send a GET request to the Web API with the question as a URL parameter The client will specify the header Content-Type: application/json in their requests as convention. A valid API query is a single URL parameter containing one sentence that is a question in standard English.

Tahap kedua adalah klasifikasi data berdasarkan tujuan kebutuhan. Dalam hal ini, data dibagi ke dalam kategori produk dan proses. Kebutuhan produk mengacu pada spesifikasi fitur atau layanan perangkat lunak yang akan digunakan oleh pengguna. Sebaliknya, kebutuhan proses berhubungan dengan aktivitas pengembangan, pemeliharaan, atau operasional perangkat lunak. Setiap data diklasifikasikan sebagai kebutuhan produk atau proses. Pengategorian produk dan proses di bawah dilakukan oleh ahli *software requirement* sebagai annotator dengan menggunakan petunjuk teknis yang sudah di tetapkan. Latar belakang Pendidikan dari annotator sebagai berikut

Tabel 3. 2 Latar Belakang Anotator

Anotator	Latar Belakang Pendidikan	Latar Belakang Pekerjaan
Anotator 1	1. Uin Maulana Malik Ibrahim Malang, S1-Teknik Informatika 2. Institut Teknologi Sepuluh Nopember, S2-Teknik Informatika	1. Analis IT PT Geomedia Sinergi (2012– 2014) 2. Pengajar Univerisitas Hasyim Asy'ari Jombang (2014- 2016) 3. Dosen Universitas Islam Lamongan (2017-2018) 4. Dosen Politeknik Negeri Malang (2019-sekarang)
Anotator 2	1. Universitas Brawijaya, S1-Teknik Informatika 2. Binar Academy UI/UX Designer	1. Nusantara Beta Studio | UI/UX Designer (2023) 2. Forum Human Capital Indonesia | HC Analytics & Dashboard (2024-sekarang)
Anotator 3	1. Uin Maulana Malik Ibrahim Malang, S1-Teknik Informatika,	1. COMPUTER LABORATORY ASSISTANT – UIN Maulana Malik Ibrahim Malang (2023) 2. DATA SCIENCE ENTHUSIASTS (DSE) – Malang, Indonesia (2022 -2023) 3. PT AIRA TECHNOLOGY INDONESIA - Malang, Indonesia (2024 – sekarang)

Petunjuk Teknis dari dari pengelompokkan kategori proses dan produk dapat diliat sebagai berikut

Tabel 3. 3 Petunjuk Teknis Pengelompokan

Kategori	Petunjuk Pengelompokan
Produk	Persyaratan yang memberi gambaran mengenai apa yang harus dilakukan suatu perangkat lunak. Ini meliputi fitur, fungsi, dan perilaku yang wajib dimiliki setiap perangkat lunak ketika dikembangkan. Contoh: "Perangkat lunak harus dapat memverifikasi bahwa seorang siswa memenuhi semua prasyarat sebelum dia mendaftar untuk suatu mata kuliah."
Proses	Persyaratan yang berkaitan dengan bagaimana perangkat lunak harus dikembangkan. Ini mencakup aturan atau metode yang harus diikuti selama pengembangan perangkat lunak. Contoh: "Perangkat lunak harus dikembangkan menggunakan proses RUP *(Rational Unified Process)*."

Dalan pengategorian untuk data produk disingkat PD dan untuk data proses dapat disingkat PS. Contoh data *software requirement* dengan kategori produk dan proses sebagai berikut

Tabel 3. 4 Contoh Data *Software Requirement* kategori produk dan proses

Type	Requirement
PD	Restaurant Owner Create Account Security GIST: The security of creating account for restaurant owners of the system. SCALE: If a restaurant owner wants to create an account and the desired user name is occupied the restaurant owner should be asked to choose a different user name. METER: Measurements obtained on 1000 hours of usage during testing. MUST: 100% of the time.
PS	System Availability GIST: The availability of the system when it is used. SCALE: The average system availability (not considering network failing). METER: Measurements obtained from 1000 hours of usage during testing. MUST: More than 98% of the time. PLAN: More than 99% of the time. WISH: 100% of the time.

Lanjutan Tabel Contoh Data *Software Requirement* kategori produk dan proses

PS	Mobile application - Search DESC: Given that a user is logged in to the mobile application then the first page that is shown should be the search page. The user should be able to search for a restaurant according to several search options. The search options are Price Destination Restaurant type and Specific dish. There should also be a free text search option. A user should be able to select multiple search options in one search.
PS	The client will send a GET request to the Web API with the question as a URL parameter The client will specify the header Content-Type: application/json in their requests as convention. A valid API query is a single URL parameter containing one sentence that is a question in standard English.

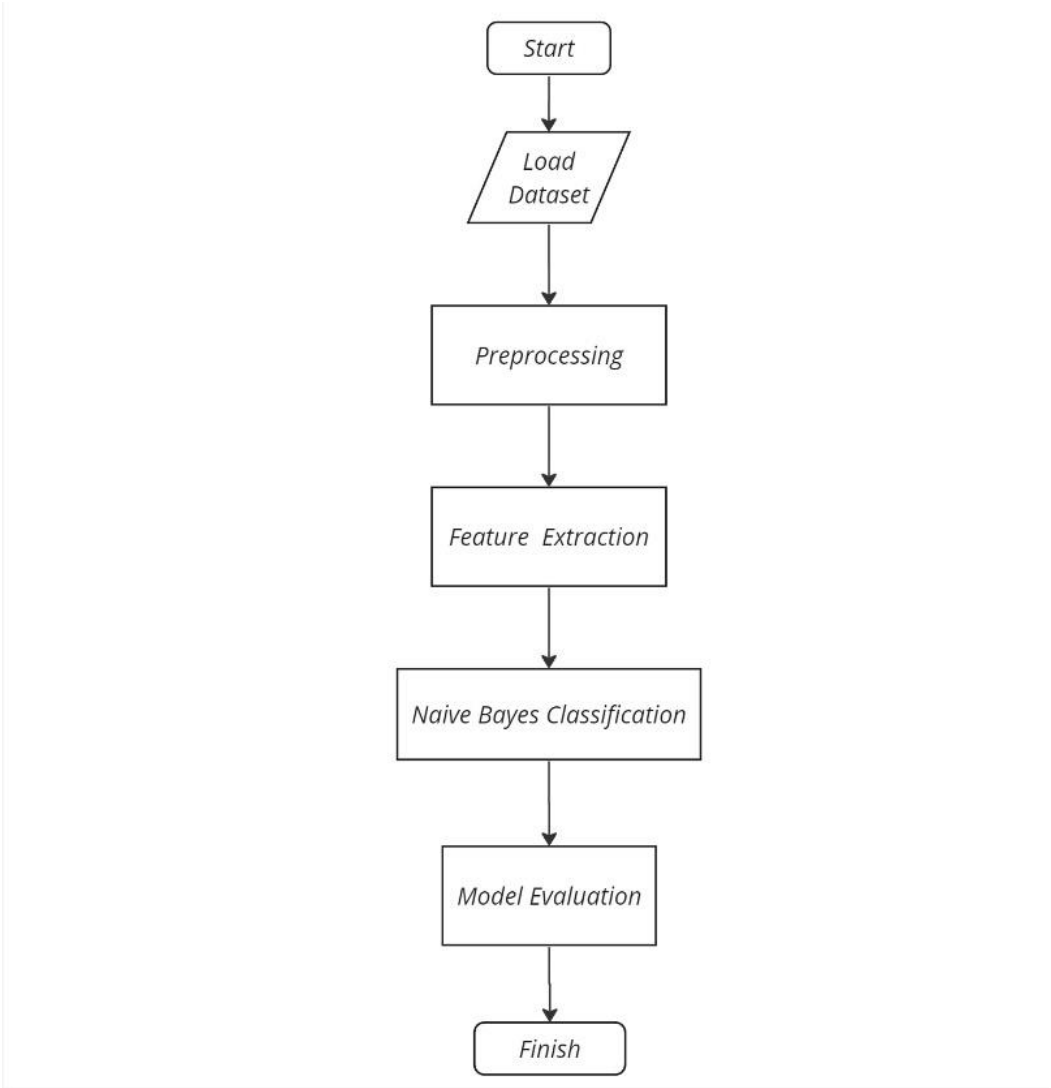
Selanjutnya, pada tahap ketiga, dilakukan klasifikasi gabungan untuk mengidentifikasi lebih spesifik hubungan antara sifat dan tujuan kebutuhan. Dari sini, terbentuk empat kategori, sebagai berikut fungsional produk, yang mencakup kebutuhan fungsional terkait fitur perangkat lunak. Fungsional Proses, yang mencakup kebutuhan fungsional dalam proses pengembangan perangkat lunak. Nonfungsional Produk, yang mencakup kebutuhan nonfungsional untuk meningkatkan kualitas fitur perangkat lunak. Nonfungsional Proses, yang mencakup kebutuhan nonfungsional dalam peningkatan kualitas proses pengembangan. Melalui pendekatan ini, setiap data dianalisis secara mendalam untuk memastikan masuknya ke salah satu dari empat kategori gabungan tersebut. Pengategorian untuk fungsional produk disingkat dengan FRPD, fungsional proses disingkat dengna FRPS, nonfungsional produk disingkat dengan NFRPD, dan nonfungsional proses disingkat dengan NFRPS. Contoh data *software requirement* dengan kategori fungsional produk, fungsional proses nonfungsional produk dan nonfungsional proses sebagai berikut

Tabel 3. 5 Contoh data *software requirement* fungsional produk, fungsional proses nonfungsional produk dan nonfungsional proses

Type	Requirement
NFRPD	Restaurant Owner Create Account Security GIST: The security of creating account for restaurant owners of the system. SCALE: If a restaurant owner wants to create an account and the desired user name is occupied the restaurant owner should be asked to choose a different user name. METER: Measurements obtained on 1000 hours of usage during testing. MUST: 100% of the time.
NFRPS	System Availability GIST: The availability of the system when it is used. SCALE: The average system availability (not considering network failing). METER: Measurements obtained FRom 1000 hours of usage during testing. MUST: More than 98% of the time. PLAN: More than 99% of the time. WISH: 100% of the time.
FRPD	Mobile application - Search DESC: Given that a user is logged in to the mobile application then the first page that is shown should be the search page. The user should be able to search for a restaurant according to several search options. The search options are Price Destination Restaurant type and Specific dish. There should also be a free text search option. A user should be able to select multiple search options in one search.
FRPS	The client will send a GET request to the Web API with the question as a URL parameter The client will specify the header Content-Type: application/json in their requests as convention. A valid API query is a single URL parameter containing one sentence that is a question in standard English.

3.2 Desain Sistem

Desain sistem adalah suatu gambaran dari sistem yang akan dibentuk dalam bentuk alur. Sistem dimulai dengan input *load dataset* berupa dokumen *Software Requirement.* Setelah itu, melakukan *text preprocessing* untuk membantu dalam merapikan dan mengambil informasi yang berguna dari data yang digunakan. Pada *text preprcessing* memiliki beberapa proses yang dilakukan diantaranya menghapus tanda baca dan karakter khusus*, case folding*, tokenisasi, *stopword, lemmatization.* Desain sistem dari penelitian ini dapat dilihat pada gambar 3.1.



Gambar 3. 1 Desain Sistem

Dalam proses *text preprocessing* didapatkan dua output yaitu data untuk uji dan data untuk latih. Setelah mendapatkan dua data, lalu dapat dilakukan proses latih yang akan membangun model dan akan dilatih oleh data latih. Dalam pelatihan data latih digunakan TF-IDF sebagai feature extraction dan penggunaan metode Naïve Bayes. Kemudian hasil dari proses *training* yang sudah dilatih akan di uji mengunakan data uji. Hasil dari proses diatas akan dibandingkan dengan data akurat untuk mengetahui keakuratan dari hasil klasifikasi.

3.3 *Text Preprocessing*

Text Preprocessing merupakan langkah dalam proses klasifikasi teks yang dilakukan sebelum pemrosesan lebih lanjut. Tujuannya adalah untuk merapikan dan mempersiapkan data teks agar lebih sesuai sehinggan mudah untuk diolah. Pada *text preprocessing* terdapat beberapa langkah *case folding*, menghapus tanda baca dan karakter khusus*,* tokenisasi, *stopword, lemmatization*.

3.3.1 Case Folding

Proses yang digunakan dalam mengubah suatu teks ke aksara kecil dapat disebut dengan *case folding*. Mengubah teks menjadi huruf kecil bertujuan agar membantu penganalisisan data tetap konsisten. Pengubahan teks menjadi huruf kecil juga membantu menghindari kesalahan dalam mencocokkan kata-kata. Contoh mengubah ke huruf kecil pada penelitian ini bisa dilihat pada tabel 3.6.

Tabel 3. 6 Contoh Penggunaan Case Folding

Sebelum	Sesudah
"The primary performance requirement is speed of the network. The application itself will only have minimal logic and so there should be little to no issues with the computation required by the phone itself."	"the primary performance requirement is speed of the network. the application itself will only have minimal logic and so there should be little to no issues with the computation required by the phone itself."
"Select English as preferred language Given the restaurant owner wants to select a preferred language When the restaurant owner selects English as a new language Then the web-portal will show all text in English."	"select english as preferred language given the restaurant owner wants to select a preferred language when the restaurant owner selects english as a new language then the web-portal will show all text in english."
"The system should be language agnostic, since portability is a prime goal of the overall project. English, turkish, Finnish, and Italian should be at least be supported because these are the replication partners."	"the system should be language agnostic, since portability is a prime goal of the overall project. english, turkish, finnish, and italian should be at least be supported because these are the replication partners."

3.3.2 Menghapus Tanda Baca dan Karakter Khusus

Kemudian dilakukan penghapusan karakter khusus dan tanda baca dilakukan memiliki tujuan agar tidak mengaburkan pola-pola dalam data, tanda baca dan

karakter khusus tidak dapat memberikan informasi yang berguna. Metode yang digunakan untuk penghapusan tanda baca yaitu '*translate()*' dan untuk penghapusan karakter khusus menggunakan metode '*string.punctuation*'. Penghapusan tanda baca dan karakter khusus dapat membentu agar proses tetap fokus pada kata-kata dan maknanya. Contoh menghapus tanda baca dan karakter khusus pada penelitian ini bisa dilihat pada tabel 3.7.

Tabel 3. 7 Contoh Penggunaan Menghapus Tanda Baca dan Karakter Khusus

Sebelum	Sesudah
"the primary performance requirement is speed of the network. the application itself will only have minimal logic and so there should be little to no issues with the computation required by the phone itself."	"the primary performance requirement is speed of the network the application itself will only have minimal logic and so there should be little to no issues with the computation required by the phone itself"
"select english as preferred language given the restaurant owner wants to select a preferred language when the restaurant owner selects english as a new language then the web-portal will show all text in english."	"select english as preferred language given the restaurant owner wants to select a preferred language when the restaurant owner selects english as a new language then the web portal will show all text in English"
"the system should be language agnostic, since portability is a prime goal of the overall project. english, turkish, finnish, and italian should be at least be supported because these are the replication partners."	"the system should be language agnostic since portability is a prime goal of the overall project english turkish finnish and italian should at least be supported because these are the replication partners"

3.3.3 Tokenisasi

Tokenisasi adalah proses memecah teks, paragraf, maupun dokumen ke dalam bentuk kata, simbol, maupun frasa yang bermakna lain yang dapat disebut dengan token. Dalam membagi teks dapat menggunakan fungsi '*words = word_tokenize(all_text)*' agar teks dapat terbagi menjadi token-token berdasarkan spasi. Tujuan penggunaan tokenisasi dalam klasifikasi teks adalah agar dapat mempersiapkan teks sehingga mudah diolah lebih lanjut. Contoh tokenisasi di penelitian ini bisa dilihat pada tabel 3.8.

Tabel 3. 8 Contoh Penggunaan Tokenisasi

Sebelum	Sesudah
“the primary performance requirement is speed of the network the application itself will only have minimal logic and so there should be little to no issues with the computation required by the phone itself”	[“the”, “primary”, “performance”, “requirement”, “is”, “speed”, “of”, “the”, “network”, “The”, “application”, “itself”, “will”, “only”, “have”, “minimal”, “logic”, “and”, “so”, “there”, “should”, “be”, “little”, “to”, “no”, “issues”, “with”, “the”, “computation”, “required”, “by”, “the”, “phone”, “itself”]
“select english as preferred language given the restaurant owner wants to select a preferred language when the restaurant owner selects english as a new language then the web portal will show all text in english"	[“select”, “english”, “as”, “preferred”, “language”, “given”, “the” “restaurant”, “owner”, “wants”, “to”, “select”, “a”, “preferred”, “language”, “when”, “the”, “restaurant”, “owner”, “selects”, “english”, “as”, “a”, “new”, “language”, “Then”, “the”, “web-portal”, “will”, “show”, “all”, “text”, “in”, “english”]
“the system should be language agnostic since portability is a prime goal of the overall project english turkish finnish and italian should at least be supported because these are the replication partners”	[“the”, “system”, “should”, “be”, “language”, “agnostic”, “since”, “portability”, “is”, “a”, “prime”, “goal”, “of”, “the”, “overall”, “project”, “english”, “turkish”, “finnish”, “and”, “italian”, “should”, “be”, “at”, “least”, “be”, “supported”, “because”, “these”, “are”, “the”, “replication”, “partners”]

3.3.4 Stopword

Stopwords dapat dilakukan penghapusan dikarenakan kosakata umum sering digunakan tidak terlalu memberikan kontribusi untuk menganalisis teks sehingga dapat menyebabkan gangguan dalam data. Dalam penghapusan *stopwords* dapat memberi fokus pada analisis kosakata yang lebih informatif. Penghapusan *stopwords* dilakukan dengan menghapus kata-kata umum yang ada di pustaka NLTL (*Natural Language Toolkit*). Contoh *stopword* pada penelitian ini bisa dilihat pada tabel 3.9.

Tabel 3. 9 Contoh Penggunaan *Stopword*

Sebelum	Sesudah
[“the”, “primary”, “performance”, “requirement”, “is”, “speed”, “of”, “the”, “network”, “The”, “application”, “itself”, “will”, “only”, “have”, “minimal”, “logic”, “and”, “so”, “there”, “should”, “be”, “little”, “to”, “no”, “issues”, “with”, “the”, “computation”, “required”, “by”, “the”, “phone”, “itself”]	[“primary”, “performance”, “requirement”, “speed”, “network”, “application”, “itself”, “only”, “minimal”, “logic”, “little”, “issues”, “computation”, “required”, “phone”, “itself”]
[“select”, “english”, “as”, “preferred”, “language”, “given”, “the” “restaurant”, “owner”, “wants”, “to”, “select”, “a”, “preferred”, “language”, “when”, “the”, “restaurant”, “owner”, “selects”, “english”, “as”, “a”, “new”, “language”, “Then”, “the”, “web-portal”, “will”, “show”, “all”, “text”, “in”, “english”]	[“select”, “english”, “preferred”, “language”, “given”, “restaurant”, “owner”, “wants”, “select”, “preferred”, “language”, “restaurant”, “owner”, “selects”, “english”, “new”, “language”, “web-portal”, “show”, “all”, “text”, “english”]
[“the”, “system”, “should”, “be”, “language”, “agnostic”, “since”, “portability”, “is”, “a”, “prime”, “goal”, “of”, “the”, “overall”, “project”, “english”, “turkish”, “finnish”, “and”, “italian”, “should”, “be”, “at”, “least”, “be”, “supported”, “because”, “these”, “are”, “the”, “replication”, “partners”]	[“system”, “language”, “agnostic”, “since”, “portability”, “prime”, “goal”, “overall”, “project”, “english”, “turkish”, “finnish”, “italian”, “least”, “supported”, “replication”, “partners”]

3.3.5 *Lemmatization*

Mengubah kata-kata menjadi bentuk dasarnya merupakan suatu proses dari *lemmatization*. Contoh *lemmatization* adalah kata "*running*" berubah menjadi kata dasar yaitu "run". Proses *lemmatization* menggunakan *WordNetLemmatizer*. Contoh *lemmatization* pada penelitian ini bisa diliat pada tabel 3.10.

Tabel 3. 10 Contoh Penggunaan *Lemmatization*

Sebelum	Sesudah
[“primary”, “performance”, “requirement”, “speed”, “network”, “application”, “itself”, “only”, “minimal”, “logic”, “little”, “issues”, “computation”, “required”, “phone”, “itself”]	["primary", "perform", "requirement", "speed", "network", "application", "it", "only", "minimal", "logic", "little", "issue", "computation", "require", "phone", "it"]
[“select”, “english”, “preferred”, “language”, “given”, “restaurant”, “owner”, “wants”, “select”, “preferred”, “language”, “restaurant”, “owner”, “selects”, “english”, “new”, “language”, “web-portal”, “show”, “all”, “text”, “english”]	["select", "english", "prefer", "language", "give", "restaurant", "owner", "want", "select", "prefer", "language", "restaurant", "owner", "select", "english", "new", "language", "web-portal", "show", "all", "text", "english"]

Lanjutan Tabel Contoh Penggunaan *Lemmatization*

[“system”, “language”, “agnostic”, “since”, “portability”, “prime”, “goal”, “overall”, “project”, “english”, “turkish”, “finnish”, “italian”, “least”, “supported”, “replication”, “partners”]	["system", "language", "agnostic", "since", "portability", "prime", "goal", "overall", "project", "english", "turkish", "finnish", "italian", "least", "support", "replication", "partner"]

3.4 *Feature Extraction*

Feature Extraction pada penelitian menggunakan pembobotan TF-IDF. *Term Frequency* (TF) dapat diartikan banyaknya kata yang dapat muncul dalam sebuah dokumen sehingga semakin banyak kata yang akan muncul maka milai dari TF besar. *Inverse Document Frequency* (IDF) dapat diartikan banyaknya data yang mengandung kata tersebut (Mayasari & Indarti, 2022). *Term Frequency-Inverse Document Frequency* (TF-IDF) dapat diartikan suatu pembobotan yang berfungsi mengubah data teks menjadi numerik. TF-IDF dapat dikatakan sebuah teknik yang digunakan dalam mengelolah teks agar kata-kata yang ada dalam dokumen mempunyai bobot (Wati et al., 2023). Tujuan dari penggunaan TF-IDF yaitu teridentifikasi kata-kata yang penting dari kumpulan data persyaratan perangkat lunka. Nilai TF dapat dihitung dengan rumus sebagai berikut :

$$TF_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{3.1}$$

dimana:
$n_{i,j}$: frekuensi kemunculan kata w_i dalam persyaratan perangkat lunak r_j
$\sum_k n_{k,j}$: total kata dalam persyaratan perangkat lunak r_j
i : Kata yang spesifik sedang dihitung frekuensinya.
j : Dokumen tertentu dalam kumpulan dokumen.
k : Semua kata dalam dokumen yang dipakai untuk menghitung total kata.

r_j adalah dokumen atau persyaratan perangkat lunak yang ke-j dalam daftar. Nilai TF tidak dapat menghitung pentingnya kata dalam data persyaratan perangkat lunak yang digunakan. Maka dari itu, dibutuhkan IDF yang dapat memberikan

bobot pada setiap kata yang jarang muncul pada data persyaratan perangkat lunak. Nilai IDF bisa dihitung dengan rumus sebagai berikut :

$$IDF_i = \log\left(\frac{N}{df_i}\right) \tag{3.2}$$

dimana :
N : jumlah total persyaratan perangkat lunak dalam dataset.
df_i : jumlah persyaratan perangkat lunak yang mengandung kata w_i.
log : logaritma natural.

Sebelum melakukan perhitungan IDF, menentukan kata-kata muncul pada satu data persyaratan atau kedua data persyaratan. Kata-kata yang digunakan di atas semua hanya muncul di satu data persyaratan saja sehingga menghasilkan hasil yang sama jika dihitung.

Selanjutnya, nilai TF-IDF bisa dihitung menggunakan cara mengalikan nilai TF dan nilai IDF. Nilai TF-IDF dapat dihitung dengan rumus sebagai berikut :

$$TF - IDF_{i,j} = TF_{i,j} \times IDF_i \tag{3.3}$$

Terakhir, mengalikan nilai TF dan nilai IDF agar mendapatkan nilai TF-IDF untuk setiap kata dalam data persyaratan yang berfungsi untuk memberikan bobot pada w_i dalam persyaratan perangkat lunak.

3.5 Metode *Naïve Bayes*

Metode *Naïve Bayes* dapat digunakan untuk setelah melakukan proses *feature extraction* dengan menggunakan TF-IDF. Langkah selanjutnya yaitu fase pelatihan. Selama fase pelatihan, setiap persyaratan perangkat lunak telah ditetapkan kategorinya masing-masing (data latih), yang nantinya akan dianalisis untuk menghasilkan pengetahuan dalam bentuk nilai probabilitas untuk setiap kata.

Proses yang akan dilakukan akan melibatkan perhitungan probabilitas dalam menentukan setiap kata dalam setiap kategori dengan menggunakan TF-IDF yang telah dilakukan pada proses sebelumnya.

Setelah melakukan perhitungan TF-IDF dilakukan perhitungan probabilitas untuk setiap kata dalam setiap kategori yang ditentukan. Dalam menghitung probabilitas dengan melibatkan penggunaan rumus perhitungan nilai TF-IDF dalam menghitung kata dan jumlah kata dalam data persyaratan perangkat lunak pada kategori tertentu. Rumus yang akan digunakan sebagai berikut :

$$p(w_i \,|c_j) = \frac{TF - IDF_{i,j} + 1}{\sum_{k=1}^{|kosakata|} TF - IDF_{k,j} + |kosakata|} \tag{3.4}$$

dimana :

$p(w_i \,|c_j)$: probabilitas kata w_i pada setiap kategori c_j

$TF - IDF_{i,j}$: nilai TF-IDF untuk kata w_i dalam dokumen c_j

$\sum_{k=1}^{|kosakata|} TF - IDF_{k,j}$: jumlah total dari TF-IDF pada semua kata dalam dokumen dalam kategori tertentu

$|kosakata|$: ukuran kosakata (jumlah kata unik dalam seluruh dataset)

Pada perhitungan probabilitas kata menggunakan penambahan 1 atau *Smoothing* untuk menghindari probabilitas nol terutama pada kata-kata yang tidak muncul dalam kategori. Selain probabilitas kata yang dihitung, probabilitas dokumen kategori juga akan dihitung. Perhitungan probabilitas persyaratan perangkat lunak kategori dapat mencerminkan proporsi persyaratan perangkat lunak yang termasuk dalam kategori tertentu dari keseluruhan data latih (Devita et al., 2018). Rumus probabilitas kategori sebagai berikut :

$$p(c_j) = \frac{n(doc_j)}{n(sampel)} \tag{3.5}$$

dimana :

$p(c_j)$: kemungkinan dokumen kategori

$n(reg_j)$: jumlah dari seluruh dokumen pada suatu kategori

$n(total)$: jumlah dari seluruh dokumen latih.

Menghitung kemungkinan kategori dapat dilakukan dengan cara menghitung jumlah dari seluruh persyaratan perangkat lunak. Setelah selesai proses pelatihan, tahap selanjutnya melibatkan prosedur kategorisasi. Dalam prosedur ini, data persyaratan perangkat lunak yang memiliki kategori yang tidak diketahui, juga disebut sebagai data uji akan digunakan. Oleh karena itu, metodologi *Naive Bayes* berusaha mengenali istilah-istilah dalam set pengujian yang sejalan dengan informasi yang diperoleh dari set pelatihan $p(w_i \mid c_j)$. Selanjutnya, probabilitas setiap persyaratan perangkat lunak $p(c_j)$ yang disimpan dalam basis pengetahuan selama proses pelatihan sebelumnya akan dihitung. Dalam perhitungan melibatkan perkalian probabilitas kata yang sesuai dengan nilai TF-IDF dengan probabilitas persyaratan perangkat lunak kategori (Devita et al., 2018). Rumus yang akan digunakan sebagai berikut :

$$p(c_j|d) \propto p(c_j)\ \Pi_i\ p(w_i\ |c_j) \tag{3.6}$$

dimana :

$p(c_j|d)$: probabilitas posterior kategori c_j pada persyaratan perangkat lunak d

$p(c_j)$: probabilitas kategori c_j dalam dataset

$p(w_i\ |c_j)$: probabilitas kata w_i pada setiap kategori persyaratan perangkat lunak c_j

$\propto$:Simbol "proporsional terhadap". Dalam konteks naive Bayes, kita tidak perlu menghitung probabilitas absolut $p(d)$ karena sama untuk semua kelas, sehingga hanya nilai relatif dari probabilitas yang digunakan untuk memutuskan kelas.

Π_i :notasi matematika yang menyatakan perkalian berulang

Kemudian, agar dapat menentukan nilai dapat melakukan suatu mengalikan nilai dari probabilitas kemunculan suatu kata yang sama dalam data latih dengan nilai kemungkinan dokumen yang sesuai dengan kategori masing-masing $p(c_j)$. Setelah mendapatkan hasil perkalian untuk setiap kategori persyaratan perangkat lunak, langkah selanjutnya adalah membandingkan serta mencari suatu nilai

probabilitas tertinggi c_{MAP} yang akan digunakan untuk mengklasifikasikan data uji untuk data *Software Requirement* ke dalam satu kategori yang sudah tersedia. Perhitungan ini bisa dilihat dalam Persamaan 4 (Devita et al., 2018).

Persamaan 4

$$c_{MAP} = argmax_{c_{j \in c}} [p(c_j)\ \Pi_i\ p(w_i\ |c_j)] \quad (3.7)$$

dimana:

c_{MAP} : kategori dokumen dengan probabilitas posterior tertinggi
$p(w_i\ |c_j)$: probabilitas kata w_i pada setiap kategori c_j
$p(c_j)$: probabilitas dokumen kategori
$c_{j \in c}$: c_j anggota himpunan kelas C
Π_i :notasi matematika yang menyatakan perkalian berulang
$argmax$:singkatan dari *argument of the maximum*, yang berarti nilai argumen (variabel input) yang memaksimalkan suatu fungsi

Dalam menghitung data uji dapat menggunakan rumus yang sama dengan menghitung probabilitas tertinggi. Hal pertama yang dilakukan yaitu menghitung masing-masing kategori pada data persyaratan perangkat lunak uji kemudian menghitung kategori probabilitas tertinggi. (Devita et al., 2018)

Jika dilijat dari hasil diatas menghitung probabilitas posterior untuk masing-masing kategori pada dokumen uji yang diberikan. Kategori dengan probabilitas tertinggi akan menjadi kategori yang dipilih sebagai hasil klasifikasi untuk dokumen uji.

Dalam mengevaluasi hasil dari perhitungan yang telah dihitung dapat dilakukan dengan cara menghitung *accuracy, recall, precision*, dan *F1-score*. Untuk rumus dalam menghitung dapat dilakukan sebagai berikut :

Rumus akurasi (*Accuracy*)

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ Instance} \quad (3.8)$$

Rumus Presisi (*Precision*)

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad (3.9)$$

Rumus *Recall (Sensitivity*)

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negative} \quad (3.10)$$

Rumus *F1-score*

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3.11)$$

3.6 Desain Eksperimen

Desain eksperimen dalam melakukan klasifikasi kebutuhan perangkat lunak melibatkan beberapa langkah yang pertama kombinasi parameter terbaik. Tujuan dari membuat kombinasi parameter terbaik yaitu untuk mengetahui kombinasi parameter terbaik yang digunakan dalam menganalisis *software requirement.* Analisis kombinasi parameter terbaik terdapat pada tabel 3.11.

Tabel 3. 11 Skenario Uji Variasi Data Latih

Jenis Parameter	**Teknik Pra-pemrosesan**	**Teknik Ekstraksi Fitur**
Multinomial Naïve Bayes	*Case Folding*, Menghapus Tanda Baca dan Karakter Khusus ,Tokenisasi, *Stopword, Lemmatization*	TF-IDF

Dalam pengujian terdapat beberapa skenario uji untuk mengetahui parameter yang dapat menghasilkan akurasi terbaik. Dalam skenario uji dilakukan pembagian data yang terdapat di tabel 3.12.

Tabel 3. 12 Skenario Uji Pembagian K-Fold

Train-Test	*K-Fold Cross Validation* k-10 k-15 k-20	Tuning Hyperparameter

Berdasarkan skenario pada tabel di atas, pengujian dilakukan dengan menggunakan eksplorasi kinerja model dengan penyesuaian nilai hyperparameter dan jumlah fitur yang bervariasi. Dalam skenario pengejian terdapat detail nilai dari tiap parameter yang akan digunakan pada masing-masing kategori.

Berdasarkan skenario uji pada tabel diatas, pengujian dapat dilakukan dengan menggunakan variasi kategori sebagai berikut

Tabel 3. 13 Skenario Uji Parameter Kategori

Parameter	Nilai
Fungsional dan Nonfungsional	k-10, k-15, k-20
Produk dan Proses	k-10, k-15, k-20
Fungsional Produk, Fungsional Proses, Nonfungsional Produk, dan Nonfungsional Proses	k-10, k-15, k-20

BAB IV

UJI COBA DAN PEMBAHASAN

4.1 Skenario Uji Coba

Skenario uji coba dapat diartikan sebagai sebuah langkah yang digunakan untuk mengatahui mengatahui tingkat kesalahan pada sistem dengan model yang akan dibangun. Dalam melakukan skenario uji, terdapat beberapa tahap yang diperlukan sebelum mendapatkan nilai kesalahan dari model yang akan diuji.

Dalam melakukan klasifikasi *software requirement* menggunakan metode Naïve Bayes dapat dilakukan beberapa tahapan. Tahapan pertama adalah menggumpulkan *dataset. Dataset* berasal dari public yanh diambil dari *Kaggle* sebanyak 100 data. Terdapat 4 kategori yang digunakan yaitu fungsional, nonfungsional, produk, proses, fungsional produk, fungsional proses, nonfungsional produk dan nonfungsional proses. Tahapan 3 adalah melakukan *preprocessing* yang terdiri dari *Case folding* bekerja dan teks diubah menjadi huruf kecil, penghapusan karakter khusus, angka, dan tanda baca dihapus. Kemudian, melakukan tokenisasi, penghapusan stopwords, dan lemmatization yang berfungsi untuk menjadikan kata menjadi kata dasar. Selain itu, melakukan *feature extraction* berupa TF-IDF.

Selanjutnya tahapan 3 adalah dataset akan dibagi menjadi dua yaitu data latih dan data uji. Pembagian data uji dan data latih menggunakan *K-Fold Cross Validation. K-Fold Cross Validation* yang digunakan yaitu k-10, k-15 dan k-20. Pembagian data uji dan data tes bisa dilihat pada tabel 4.1 berikut

Tabel 4. 1 Pembagian Data dalam Pengujian

Dataset	K-Fold	Data Latih	Data Uji
A	k-10	90%	10%
B	k-15	93.33%	6.67%
C	k-20	95%	5%

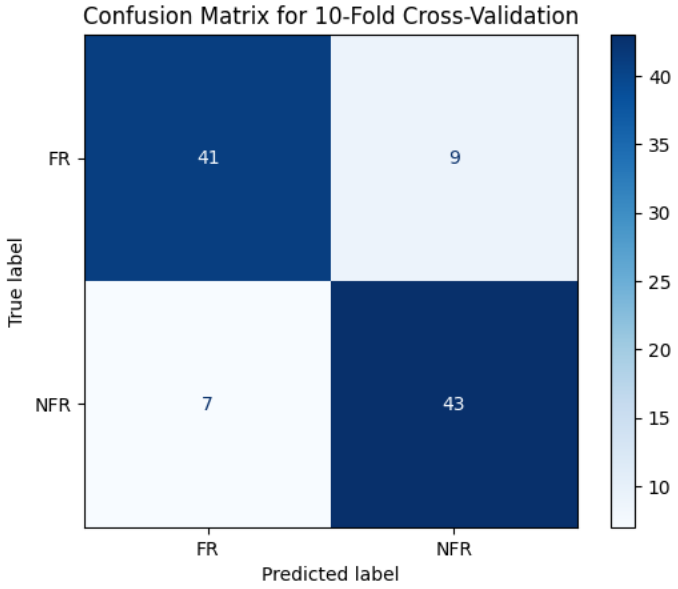
Tahapan 4 adalah melatih model Naïve Bayes menggunakan data latih. Naïve Bayes yang digunakan yaitu Multinomial Naïve Bayes. Kemudian, tahapan 5 adalah menguji model Multinominal Naïve Bayes yang telah dilatih dengan data uji untuk mengevaluasi model dalam mengklasifikasikan *software requirement.* Evaluasi model mencakup perhitungan nilai *accuracy, precision*, *recall* dan *F1-Score*. Tahapan 6 adalah Menganalisis dan membandingkan performa model *Naïve Bayes* dengan berbagai skenario pembagian data, skenario variasi kategori untuk mendapatkan kesimpulan model *Naïve Bayes* terbaik untuk mengklasifikasikan *software requirement*.

4.2 Hasil Uji Coba

Sesuai skenario pengujian, penelitian akan dilakukan 3 kali uji coba dengan menggunakan kategori pertama fungsional dan nonfungsional, kategori kedua yaitu produk dan proses serta kategori ke tiga yaitu fungsional produk, fungsional proses, nonfungsional produk, dan nonfungsional proses dengan membagi dataset menjadi beberapa rasio. Tujuan dari skenario yaitu untuk menemukan rasio yang tepat yang dapat memberikan hasil optimal dalam model klasifikasi. Model klasifikasi akan menghasilkan evaluasi performa menggunakan parameter *accuracy, recall, precision* dan *F1-Score.*

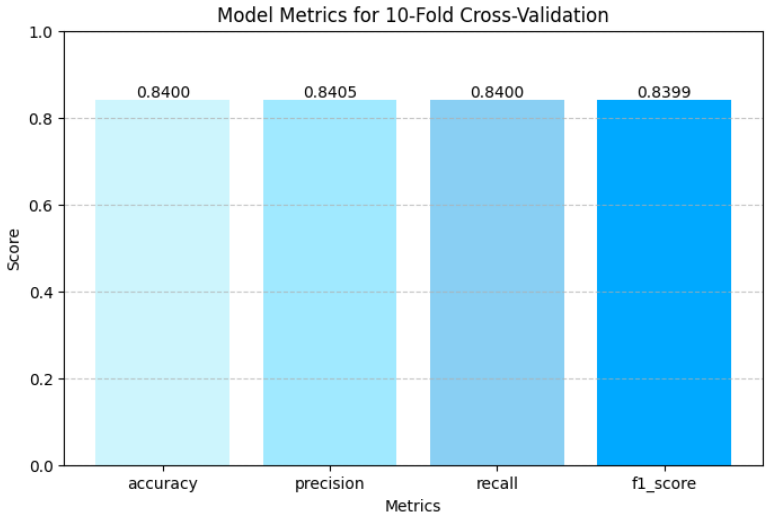
4.2.1 Hasil Uji Coba Skenario-1

Pada uji coba di skenario 1 melakukan dengan menggunakan kategori pertama yaitu fungsional dan nonfungsional menggunakan *K-Fold Cross Validation* terdiri dari k-10, k-15, dan k-20. Uji coba *K-Fold* pertama yang dilakukan ialah k-10. Berikut Gambar 4.1 merupakan hasil evaluasi pengukuran performa model menggunakan *confusion matrix* pada uji coba skenario ini.



Gambar 4. 1 *Confusion matrix* Uji Coba Skenario- 1 k-10

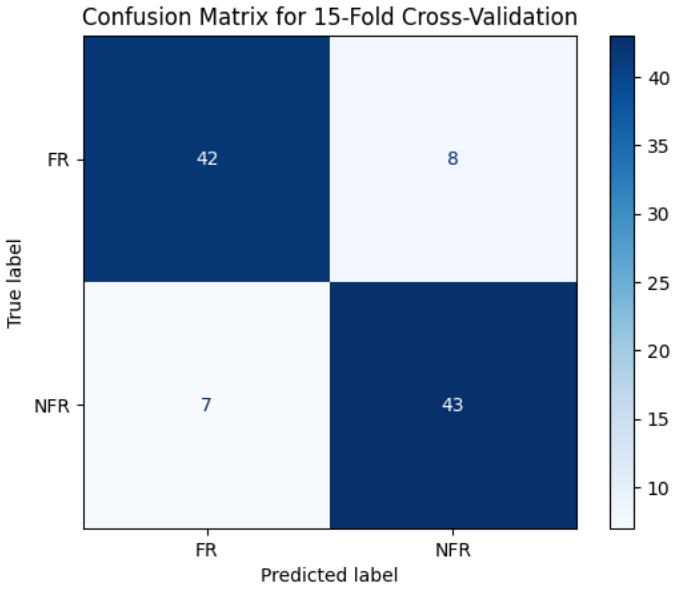
Pada gambar 4.1 di atas menunjukkan hasil evaluasi performa model klasifikasi menggunakan *confusion matrix* pada skenario penelitian yang dilakukan. Dari hasil klasifikasi, diketahui bahwa model berhasil mengklasifikasikan 41 data dengan tepat sebagai kelas fungsional, dan 43 data diklasifikasikan dengan benar sebagai kelas nonfungsional. Namun, terdapat 9 data yang seharusnya masuk ke kelas fungsional tetapi diprediksi oleh model sebagai kelas nonfungsional, dan 7 data yang seharusnya diklasifikasikan sebagai kelas nonfungsional malah diprediksi sebagai kelas fungsional.



Gambar 4. 2 Metrik Performa Uji Coba Skenario-1 k-10

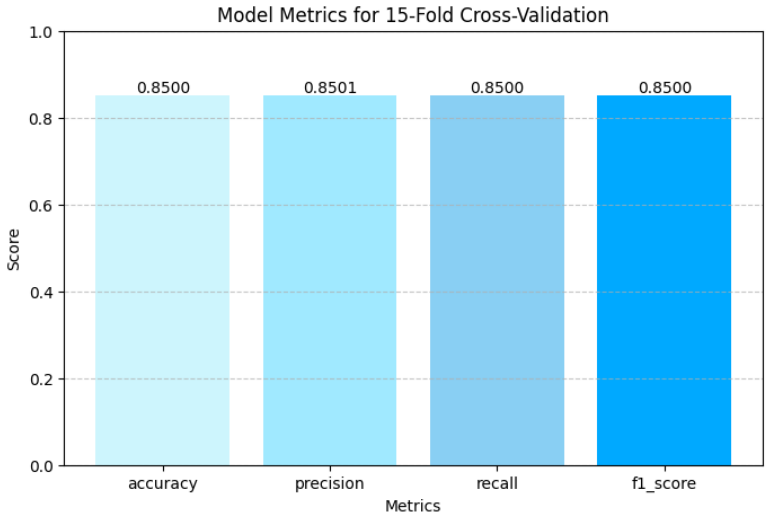
Akurasi model tercatat sebesar 0.8400, yang menunjukkan bahwa sekitar 84% dari seluruh prediksi yang dibuat oleh model. Metrik Presisi menunjukkan nilai 0.8405. Ini berarti bahwa dari semua prediksi yang dibuat oleh model sebesar 84%. *Recall* tercatat sebesar 0.8400, mengindikasikan bahwa model mengidentifikasi 84% dari semua data. Terakhir, *F1-Score* model berada di angka 0.8399.

Uji coba *K-Fold* kedua yang dilakukan ialah k-15. Berikut Gambar 4.3 merupakan hasil evaluasi pengukuran performa model menggunakan *confusion matrix* pada uji coba skenario ini.



Gambar 4. 3 *Confusion matrix* Uji Coba Skenario-1 k-15

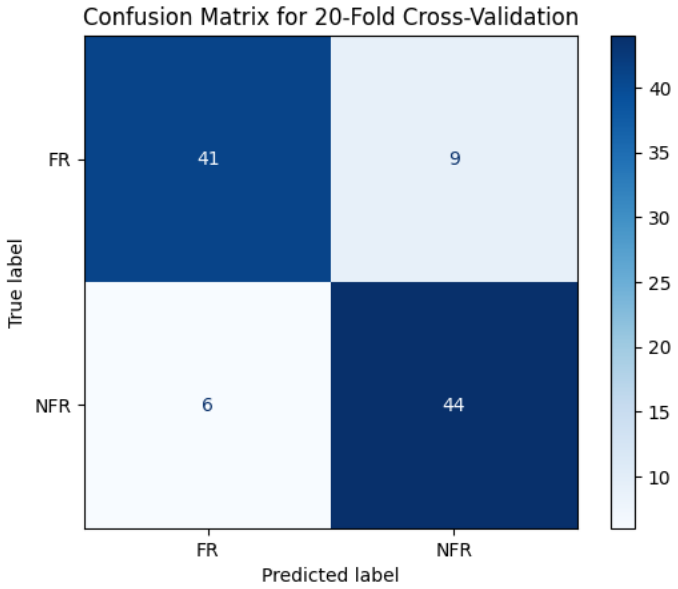
Pada gambar 4.3 yang ditunjukkan di atas mendapatkan hasil evaluasi performa model klasifikasi menggunakan *confusion matrix* pada skenario penelitian yang dilakukan. Dari hasil klasifikasi, diketahui bahwa model berhasil mengklasifikasikan 42 data dengan tepat sebagai kelas fungsional, dan 43 data diklasifikasikan dengan benar sebagai kelas nonfungsional. Namun, terdapat 8 data seharusnya masuk ke kelas fungsional namun diprediksi oleh model pada kelas nonfungsional, dan 7 data yang seharusnya diklasifikasikan sebagai kelas nonfungsional malah diprediksi sebagai kelas fungsional.



Gambar 4. 4 Metrik Performa Uji Coba Skenario-1 k-15

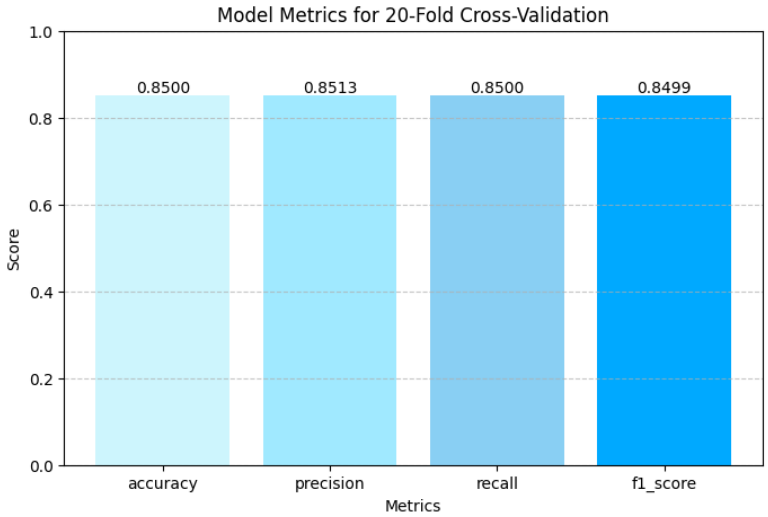
Akurasi model tercatat sebesar 0.8500, yang menunjukkan bahwa sekitar 85% dari seluruh prediksi yang dibuat oleh model. Metrik Presisi menunjukkan nilai 0.8501. Ini berarti bahwa dari semua prediksi yang dibuat oleh model sebesar 85%. *Recall* tercatat sebesar 0.8500, mengindikasikan bahwa model mengidentifikasi 85% dari semua data. Terakhir, *F1-Score* model berada di angka 0.8500.

Uji coba *K-Fold* kedua yang dilakukan ialah k-20. Berikut Gambar 4.5 merupakan hasil evaluasi pengukuran performa model menggunakan *confusion matrix* pada uji coba skenario ini.



Gambar 4. 5 *Confusion matrix* Uji Coba Skenario-1 k-20

Pada gambar 4.5 yang ditunjukkan di atas mendapatkan hasil evaluasi performa model klasifikasi menggunakan *confusion matrix* pada skenario penelitian yang dilakukan. Dari hasil klasifikasi, diketahui bahwa model berhasil mengklasifikasikan 41 data dengan tepat sebagai kelas fungsional, dan 44 data diklasifikasikan dengan benar sebagai kelas nonfungsional. Namun, terdapat 9 data yang seharusnya masuk ke kelas fungsional tetapi diprediksi oleh model sebagai kelas nonfungsional, dan 6 data yang seharusnya diklasifikasikan sebagai kelas nonfungsional malah diprediksi sebagai kelas fungsional.

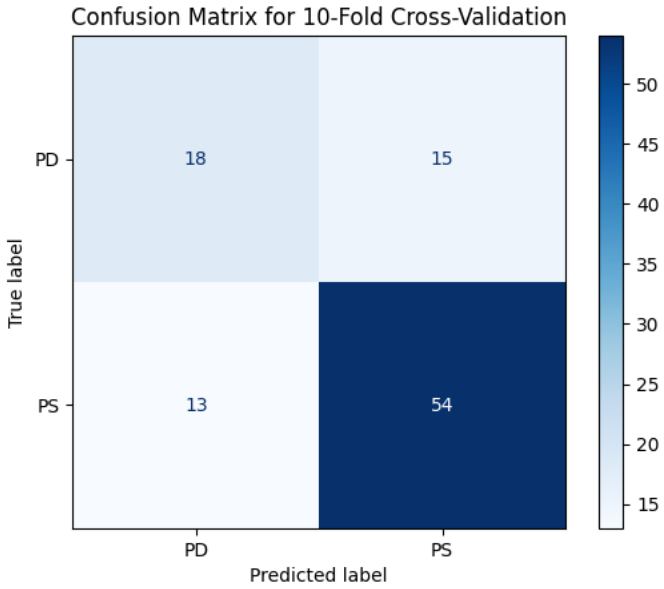


Gambar 4. 6 Metrik Performa Uji Coba Skenario-1 k-20

Akurasi model tercatat sebesar 0.8500, yang menunjukkan bahwa sekitar 85% dari seluruh prediksi yang dibuat oleh model. Metrik Presisi menunjukkan nilai 0.8513. Ini berarti bahwa dari semua prediksi yang dibuat oleh model sebesar 85%. *Recall* tercatat sebesar 0.8500, mengindikasikan bahwa model mengidentifikasi 85% dari semua data. Terakhir, *F1-Score* model berada di angka 0.8499.

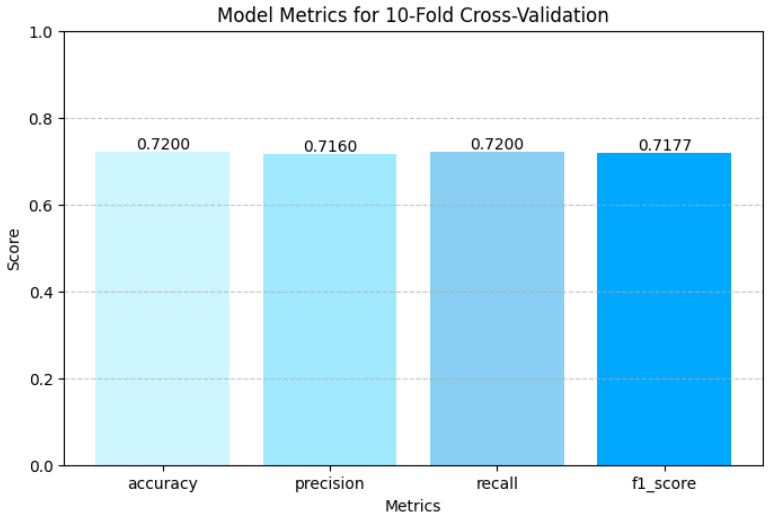
4.2.2 Hasil Uji Coba Skenario-2

Pada uji coba skenario 2 melakukan dengan menggunakan kategori pertama yaitu produk dan proses menggunakan *K-Fold Cross Validation* yang terdiri k-10, k-15, dan k-20. Uji coba *K-Fold* pertama yang dilakukan ialah k-10. Berikut Gambar 4.7 merupakan hasil evaluasi pengukuran performa model menggunakan *confusion matrix* pada uji coba skenario ini.



Gambar 4. 7 *Confusion matrix* Uji Coba Skenario-2 k-10

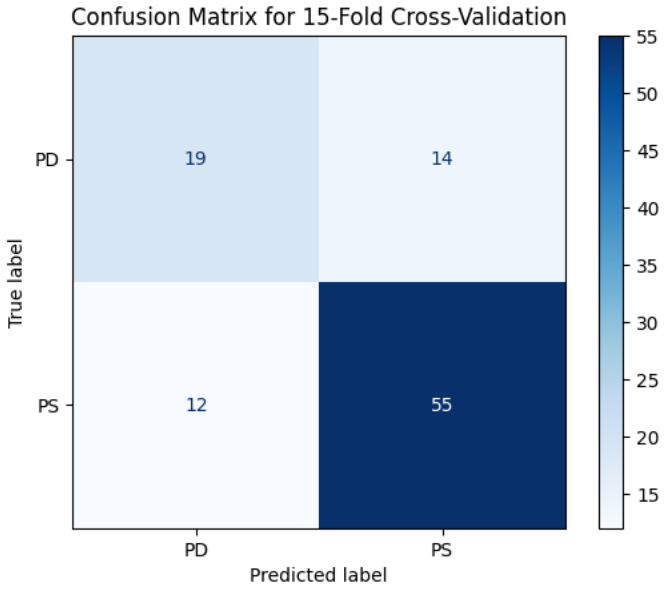
Pada gambar 4.7 yang ditunjukkan di atas mendapatkan hasil evaluasi performa model klasifikasi menggunakan *confusion matrix* pada skenario penelitian yang dilakukan. Dari hasil klasifikasi, diketahui bahwa model berhasil mengklasifikasikan 18 data dengan tepat sebagai kelas produk, dan 54 data diklasifikasikan dengan benar sebagai kelas proses. Namun, terdapat 15 data yang seharusnya masuk ke kelas produk tetapi diprediksi oleh model sebagai kelas proses, dan 13 data yang seharusnya diklasifikasikan sebagai kelas proses malah diprediksi sebagai kelas produk.



Gambar 4. 8 *Metrics Performa* Uji Coba Skenario-2 k-10

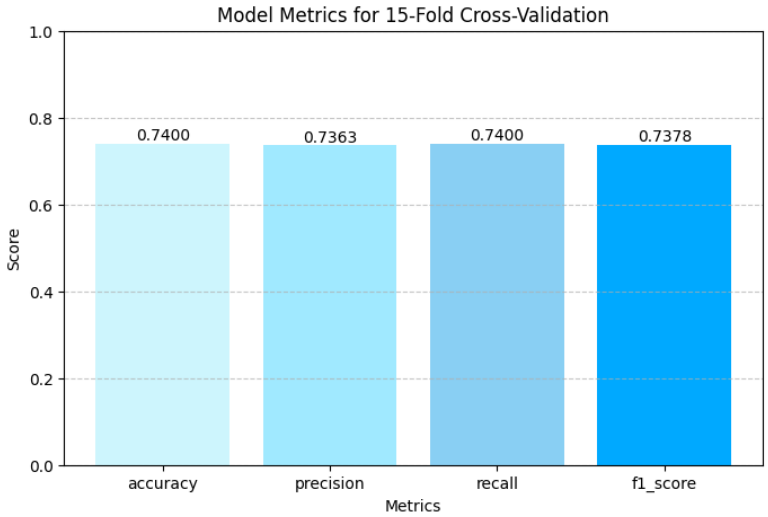
Akurasi tercatat sebesar 0.7200, yang menunjukkan bahwa sekitar 72% dari seluruh prediksi yang dibuat oleh model. Metrik Presisi menunjukkan nilai 0.7160. Ini berarti bahwa dari semua prediksi yang dibuat oleh model sebesar 71,6%. *Recall* tercatat sebesar 0.7200, mengindikasikan bahwa model mengidentifikasi 72% dari semua data. Terakhir, *F1-Score* model berada di angka 0.7177.

Uji coba *K-Fold* kedua yang dilakukan ialah k-15. Berikut Gambar 4.9 merupakan hasil evaluasi pengukuran performa model menggunakan *confusion matrix* pada uji coba skenario ini.



Gambar 4. 9 *Confusion matrix* Uji Coba Skenario-2 k-15

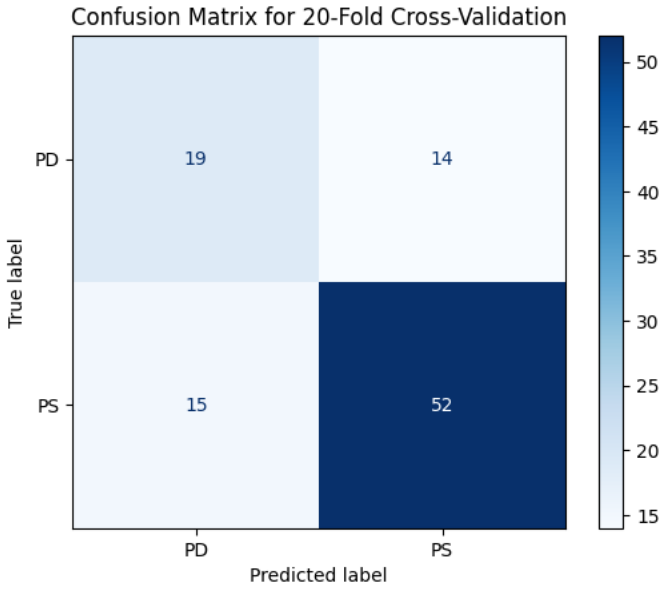
Pada gambar 4.9 di atas menunjukkan hasil evaluasi performa model klasifikasi menggunakan *confusion matrix* pada skenario penelitian yang dilakukan. Dari hasil klasifikasi, diketahui bahwa model berhasil mengklasifikasikan 19 data dengan tepat sebagai kelas produk, dan 55 data diklasifikasikan dengan benar sebagai kelas proses. Namun, terdapat 14 data yang seharusnya masuk ke kelas produk tetapi diprediksi oleh model sebagai kelas proses, dan 12 data yang seharusnya diklasifikasikan sebagai kelas proses malah diprediksi sebagai kelas produk.



Gambar 4. 10 *Metrics Performa* Uji Coba Skenario-2 k-15

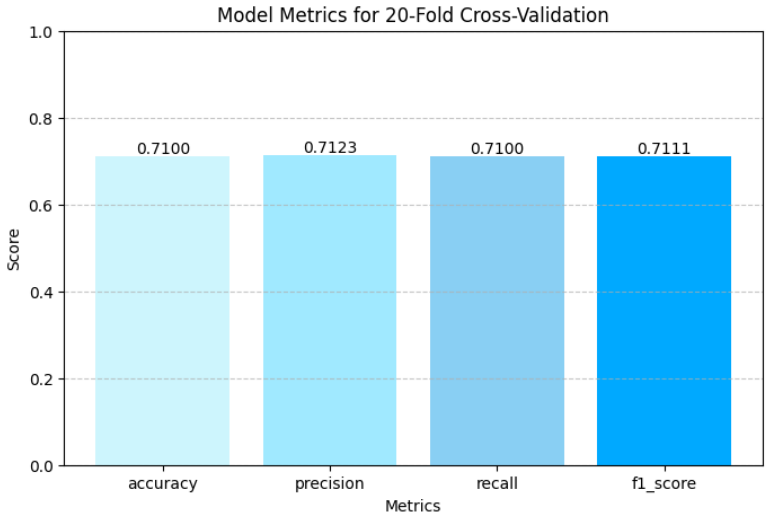
Akurasi model tercatat sebesar 0.7400, yang menunjukkan bahwa sekitar 74% dari seluruh prediksi yang dibuat oleh model. Metrik Presisi menunjukkan nilai 0.7363. Ini berarti bahwa dari semua prediksi yang dibuat oleh model sebesar 83,6%. *Recall* tercatat sebesar 0.7400, mengindikasikan bahwa model mengidentifikasi 74% dari semua data. Terakhir, *F1-Score* model berada di angka 0.7378.

Uji coba *K-Fold* ketiga yang dilakukan ialah k-20. Berikut Gambar 4.11 merupakan hasil evaluasi pengukuran performa model menggunakan *confusion matrix* pada uji coba skenario ini.



Gambar 4. 11 *Confusion matrix* Uji Coba Skenario-2 k-20

Pada gambar 4.11 di atas menunjukkan hasil evaluasi performa model klasifikasi menggunakan *confusion matrix* pada skenario penelitian yang dilakukan. Dari hasil klasifikasi, diketahui bahwa model berhasil mengklasifikasikan 19 data dengan tepat sebagai kelas produk, dan 52 data diklasifikasikan dengan benar sebagai kelas proses. Namun, terdapat 14 data yang seharusnya masuk ke kelas produk tetapi diprediksi oleh model sebagai kelas proses, dan 15 data yang seharusnya diklasifikasikan sebagai kelas proses malah diprediksi sebagai kelas produk.

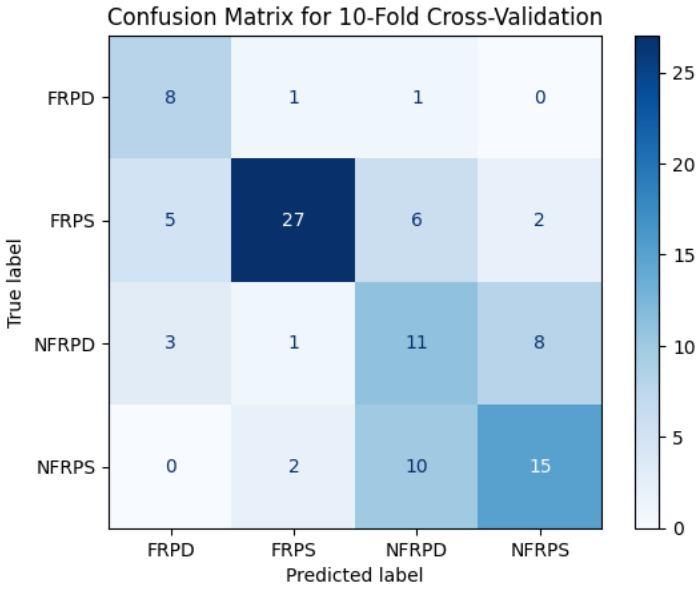


Gambar 4. 12 *Metrics Performa* Uji Coba Skenario-2 k-20

Akurasi model tercatat sebesar 0.7100, yang menunjukkan bahwa sekitar 71% dari seluruh prediksi yang dibuat oleh model. Metrik Presisi menunjukkan nilai 0.7123. Ini berarti bahwa dari semua prediksi yang dibuat oleh model sebesar 71,2%. *Recall* tercatat sebesar 0.7100, mengindikasikan bahwa model mengidentifikasi 71% dari semua data. Terakhir, *F1-Score* model berada di angka 0.7111.

4.2.3 Hasil Uji Coba Skenario-3

Pada uji coba skenario 3 melakukan dengan menggunakan kategori pertama yaitu fungsional produk, fungsional proses, nonfungsional produk dan nonfungsional proses menggunakan *K-Fold Cross Validation* yang terdiri k-10, k-15, dan k-20. Uji coba *K-Fold* pertama yang dilakukan ialah k-10. Berikut Gambar 4.13 merupakan hasil evaluasi pengukuran performa model menggunakan *confusion matrix* pada uji coba skenario ini.

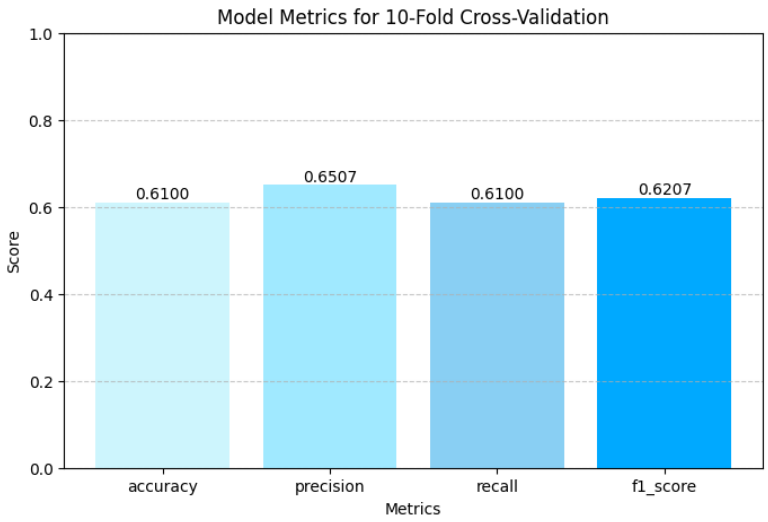


Gambar 4. 13 *Confusion matrix* Uji Coba Skenario-3 k-10

Pada gambar 4.13 di atas menunjukkan hasil evaluasi performa model klasifikasi menggunakan *confusion matrix* pada skenario penelitian dengan empat kelas, yaitu FRPD (Fungsional Produk), FRPS (Fungsional Proses), NFRPD (Nonfungsional Produk), dan NFRPS (Nonfungsional Proses). *Confusion matrix* ini memberikan gambaran mengenai kemampuan model dalam mengklasifikasikan data ke kelas yang sesuai.

Dari hasil evaluasi, dapat dijelaskan bahwa model berhasil mengklasifikasikan 8 data dengan benar ke dalam kelas FRPD, tetapi terdapat kesalahan di mana 5 data di kelas FRPS, 3 data di kelas NFRPD, serta tidak ada data di kelas NFRPS yang salah diprediksi sebagai FRPD. Untuk kelas FRPS, model menunjukkan performa yang cukup baik dengan mengklasifikasikan 27 data dengan benar. Namun, masih terdapat 1 data di kelas FRPD, 6 data di kelas NFRPD, serta 2 data di kelas NFRPS yang salah diprediksi sebagai FRPS.

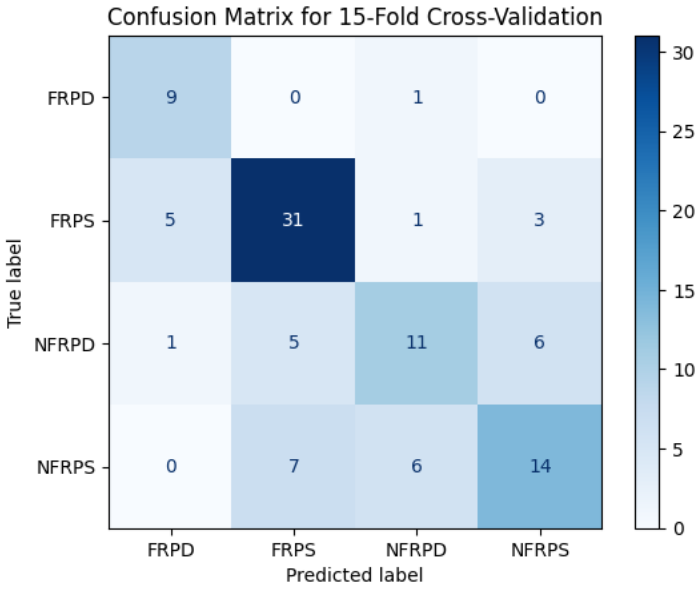
Sementara itu, untuk kelas NFRPD, model berhasil mengklasifikasikan 11 data dengan benar, tetapi masih ada 1 data di kelas FRPD, 1 data di kelas FRPS, dan 10 data di kelas NFRPS yang salah diprediksi sebagai NFRPD. Pada kelas NFRPS, model mampu mengklasifikasikan 15 data dengan tepat, tetapi terdapat 2 data dari kelas FRPS dan 8 data dari kelas NFRPD yang salah diprediksi sebagai NFRPS.



Gambar 4. 14 *Metrics Performa* Uji Coba Skenario-3 k-10

Akurasi model tercatat sebesar 0.6100, yang menunjukkan bahwa sekitar 61% dari seluruh prediksi yang dibuat oleh model. Metrik Presisi menunjukkan nilai 0.6507. Ini berarti bahwa dari semua prediksi yang dibuat oleh model sebesar 65%. *Recall* tercatat sebesar 0.6100, mengindikasikan bahwa model mengidentifikasi 61% dari semua data. Terakhir, *F1-Score* model berada di angka 0.6207.

Uji coba *K-Fold* kedua yang dilakukan ialah k-15. Berikut Gambar 4.15 merupakan hasil evaluasi pengukuran performa model menggunakan *confusion matrix* pada uji coba skenario ini.

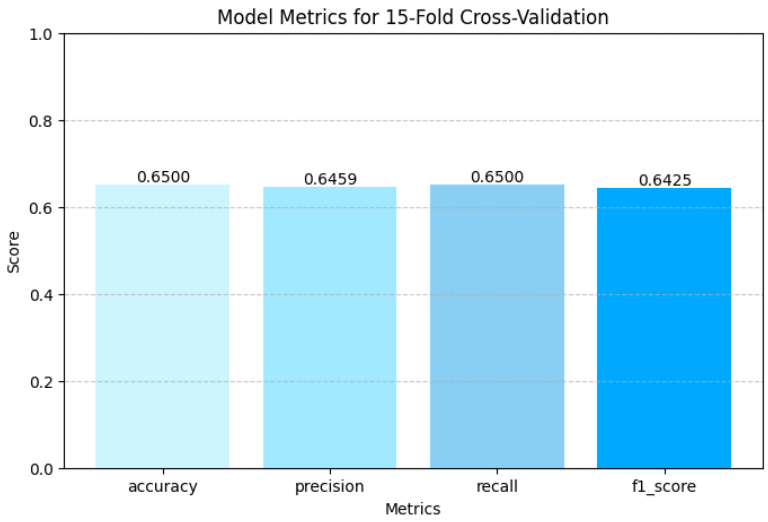


Gambar 4. 15 *Confusion matrix* Uji Coba Skenario-3 k-15

Pada gambar 4.15 di atas menunjukkan hasil evaluasi performa model klasifikasi menggunakan *confusion matrix* pada skenario penelitian dengan empat kelas, yaitu FRPD (Fungsional Produk), FRPS (Fungsional Proses), NFRPD (Nonfungsional Produk), dan NFRPS (Nonfungsional Proses). Evaluasi ini dilakukan menggunakan metode validasi silang sebanyak 15 lipatan (15-fold cross-validation).

Berdasarkan hasil evaluasi, dapat dijelaskan bahwa model berhasil mengklasifikasikan 9 data dengan benar ke dalam kelas FRPD, namun terdapat kesalahan di mana 5 data dari kelas FRPS, 1 data dari kelas NFRPD, dan tidak ada data dari kelas NFRPS salah diprediksi sebagai FRPD. Untuk kelas FRPS, model menunjukkan kinerja yang cukup baik dengan mengklasifikasikan 31 data dengan benar. Namun, masih terdapat 1 data dari kelas FRPD, 5 data dari kelas NFRPD, dan 7 data dari kelas NFRPS yang salah diprediksi sebagai FRPS.

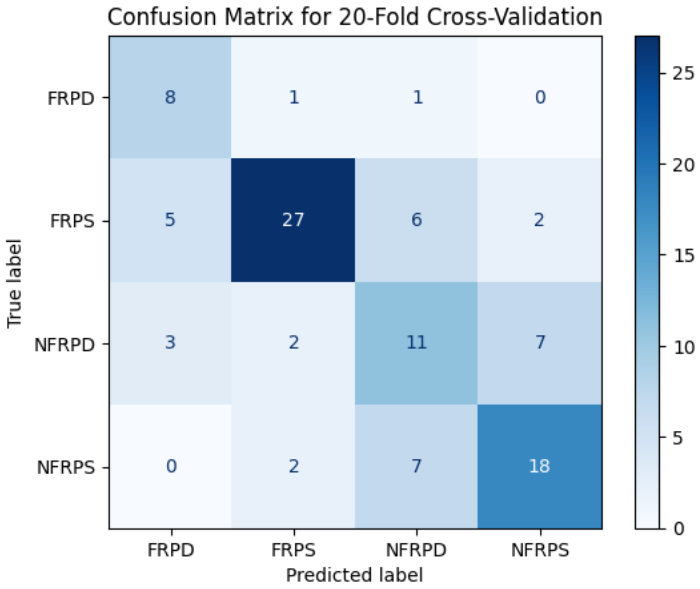
Pada kelas NFRPD, model berhasil mengklasifikasikan 11 data dengan benar. Namun, terdapat 1 data dari kelas FRPD, 5 data dari kelas FRPS, dan 6 data dari kelas NFRPS yang salah diprediksi sebagai NFRPD. Sedangkan pada kelas NFRPS, model mampu mengklasifikasikan 14 data dengan tepat, tetapi terdapat 3 data dari kelas FRPS, 6 data dari kelas NFRPD, dan tidak ada data dari kelas FRPD yang salah diprediksi sebagai NFRPS.



Gambar 4. 16 *Metrics Performa* Uji Coba Skenario-3 k-15

Akurasi model tercatat sebesar 0.6500, yang menunjukkan bahwa sekitar 65% dari seluruh prediksi yang dibuat oleh model. Metrik Presisi menunjukkan nilai 0.6459. Ini berarti bahwa dari semua prediksi yang dibuat oleh model sebesar 64,6%. *Recall* tercatat sebesar 0.6500, mengindikasikan bahwa model mengidentifikasi 65% dari semua data. Terakhir, *F1-Score* model berada di angka 0.6425.

Uji coba *K-Fold* ketiga yang dilakukan ialah k-20. Berikut Gambar 4.17 merupakan hasil evaluasi pengukuran performa model menggunakan *confusion matrix* pada uji coba skenario ini.

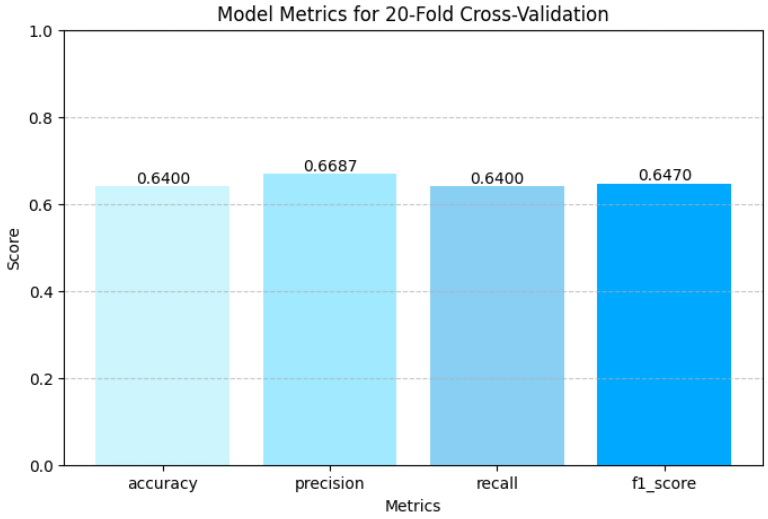


Gambar 4. 17 *Confusion matrix* Uji Coba Skenario-3 k-20

Pada gambar 4.17 di atas menunjukkan hasil evaluasi performa model klasifikasi menggunakan *confusion matrix* pada skenario penelitian dengan empat kelas, yaitu FRPD (Fungsional Produk), FRPS (Fungsional Proses), NFRPD (Nonfungsional Produk), dan NFRPS (Nonfungsional Proses). Evaluasi dilakukan menggunakan metode validasi silang sebanyak 20 lipatan (20-fold cross-validation).

Dari hasil evaluasi, diketahui bahwa model mampu mengklasifikasikan 8 data dengan benar sebagai kelas FRPD. Namun, terdapat beberapa kesalahan, yaitu 5 data di kelas FRPS, 3 data di kelas NFRPD, dan tidak ada data di kelas NFRPS yang salah diprediksi sebagai kelas FRPD. Pada kelas FRPS, model menunjukkan kinerja yang cukup baik dengan mengklasifikasikan 27 data dengan benar. Meskipun demikian, terdapat 1 data di kelas FRPD, 6 data di kelas NFRPD, dan 2 data di kelas NFRPS yang salah diprediksi sebagai kelas FRPS.

Selanjutnya, untuk kelas NFRPD, model berhasil mengklasifikasikan 11 data dengan tepat. Namun, terdapat kesalahan pada 3 data dari kelas FRPD, 6 data dari kelas FRPS, dan 7 data dari kelas NFRPS yang salah diprediksi sebagai kelas NFRPD. Sementara itu, pada kelas NFRPS, model mampu mengklasifikasikan 18 data dengan benar. Namun, terdapat 2 data di kelas FRPS, 7 data di kelas NFRPD, serta tidak ada data di kelas FRPD yang salah diprediksi sebagai kelas NFRPS.



Gambar 4. 18 *Metrics Performa* Uji Coba Skenario-3 k-20

Akurasi model tercatat sebesar 0.6400, yang menunjukkan bahwa sekitar 64% dari seluruh prediksi yang dibuat oleh model. Metrik Presisi menunjukkan nilai 0.6687. Ini berarti bahwa dari semua prediksi yang dibuat oleh model sebesar 66,9%. *Recall* tercatat sebesar 0.6400, mengindikasikan bahwa model mengidentifikasi 64% dari semua data. Terakhir, *F1-Score* model berada di angka 0.6470.

4.3 Pembahasan

Berdasrkan uji coba yang dilakukan di atas, diperoleh hasil perfroma dari 3 uji coba model klasifikasi yang terdiri 2 atau 4 kategori dengan masing-masing uji coba terdapat 3 k-fold diantaranya k-10, k-15 dan k-20. Hasil keseluruhan uji coba pertama yang bisa dilihat pada Tabel 4.2. Tabel tersebut berisi rata-rata dari split data yang telah dievaluasi menggunakan *confusion matrix* menggunakan 4 parameter yaitu *accuracy, precision, F1-Score*, dan *recall.* Parameter yang dipakai ini bertujuan untuk menilai sejauh mana kualitas model mengklasifikasikan data ulasan pada setiap uji coba.

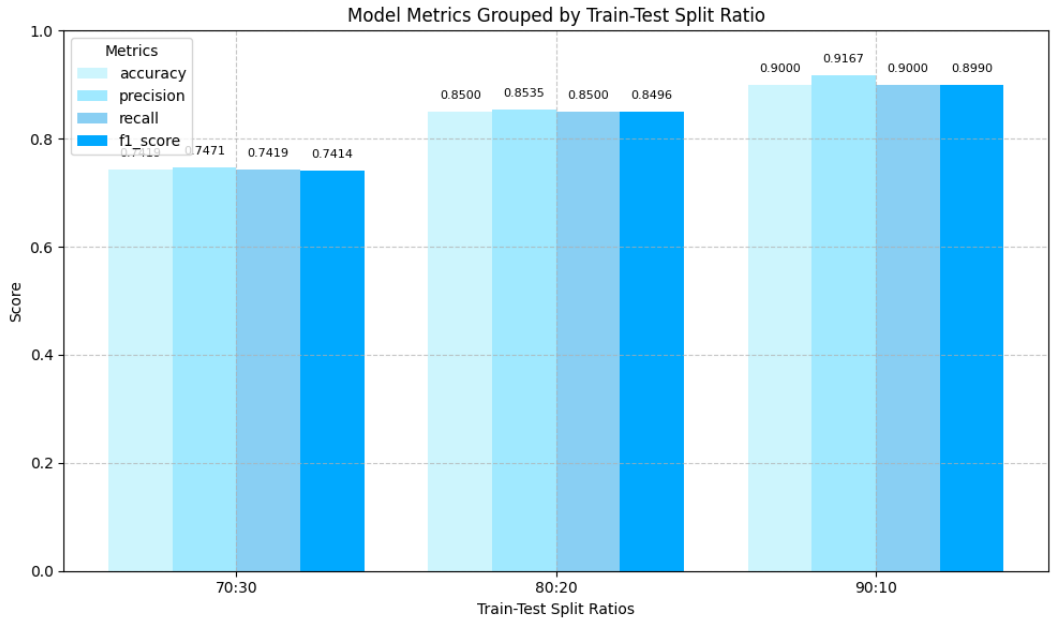
Tabel 4. 2 Hasil Uji Coba Skenario 1

Uji Coba	Variasi Kategori	K-Fold Cross Validation	*Accuracy*	*Precision*	*Recall*	*F1-Score*
1	Fungsional Nonfungsional	k-10	84%	84%	84%	84%
		k-15	85%	85%	85%	85%
		k-20	85%	85%	85%	85%

Pengujian pada kategori *Fungsional* dan *Nonfungsional* menunjukkan hasil yang cukup konsisten melalui pendekatan *K-Fold Cross Validation* menggunakan variasi jumlah lipatan (k). Pada pengujian menggunakan k-10, model berhasil mencapai *Accuracy* sebesar 84%, dengan *F1-Score* yang juga berada di angka 84%. Hasil ini mencerminkan performa model yang seimbang dalam mengklasifikasikan data dengan tingkat akurasi yang baik.

Selanjutnya, ketika jumlah lipatan ditingkatkan menjadi k-15, performa model mengalami peningkatan kecil namun berarti. Semua metrik *Accuracy* dan *F1-Score*—naik menjadi 85%. Hal ini menunjukkan bahwa penggunaan lebih banyak lipatan dalam validasi membantu model menghasilkan prediksi yang lebih baik, sehingga meningkatkan kemampuan generalisasi.

Pada k-20, performa model tetap stabil di angka 85% untuk semua metrik. Konsistensi ini menegaskan bahwa peningkatan jumlah lipatan tidak menimbulkan perubahan yang signifikan, yang mengindikasikan bahwa model sudah cukup optimal dan tidak terlalu bergantung pada variasi jumlah lipatan dalam validasi.



Gambar 4. 19 *Metrics Performa* Uji Coba Skenario 1

Pada gambar 4.19 di atas menunjukkan perbandingan performa model berdasarkan *k-Fold Cross-Validation* menggunakan nilai k yang berbeda, yaitu 10, 15, dan 20. Evaluasi dilakukan menggunakan empat metrik utama yaitu *accuracy* dan *F1-Score*. Hasil analisis menunjukkan bahwa performa model cenderung stabil di semua nilai k, dengan sedikit peningkatan pada beberapa metrik saat jumlah lipatan bertambah.

Pada k-Fold dengan nilai k-10, model mencapai skor 0.84 untuk semua metrik. Ini menunjukkan performa yang cukup baik dan seimbang tanpa adanya metrik yang secara signifikan lebih rendah dari yang lain. Namun, saat jumlah

lipatan meningkat menjadi k-15, terjadi sedikit peningkatan pada semua metrik, dengan skor meningkat menjadi 0.85. Hal ini mengindikasikan bahwa validasi silang dengan lebih banyak lipatan mampu memberikan hasil yang lebih optimal karena model diuji pada lebih banyak variasi data.

Ketika nilai k ditingkatkan lagi menjadi 20, performa model tetap stabil dengan semua metrik bertahan di angka 0.85. Ini menunjukkan bahwa meskipun jumlah lipatan bertambah, dampaknya terhadap performa model menjadi minimal. Stabilitas ini mengindikasikan bahwa model mampu melakukan generalisasi dengan efektif dan tidak terlalu terpengaruh oleh perubahan jumlah lipatan pada validasi silang.

Secara keseluruhan, grafik ini menunjukkan bahwa model mencapai performa terbaiknya pada k-Fold dengan nilai k-15 dan 20, meskipun perbedaannya dengan k-10 tidak terlalu signifikan. Hal ini menunjukkan bahwa model yang baik terhadap perubahan jumlah lipatan, namun penggunaan k-15 atau k-20 dapat dipertimbangkan untuk mendapatkan hasil yang sedikit lebih optimal.

Tabel 4. 3 Hasil Uji Coba Skenario 2

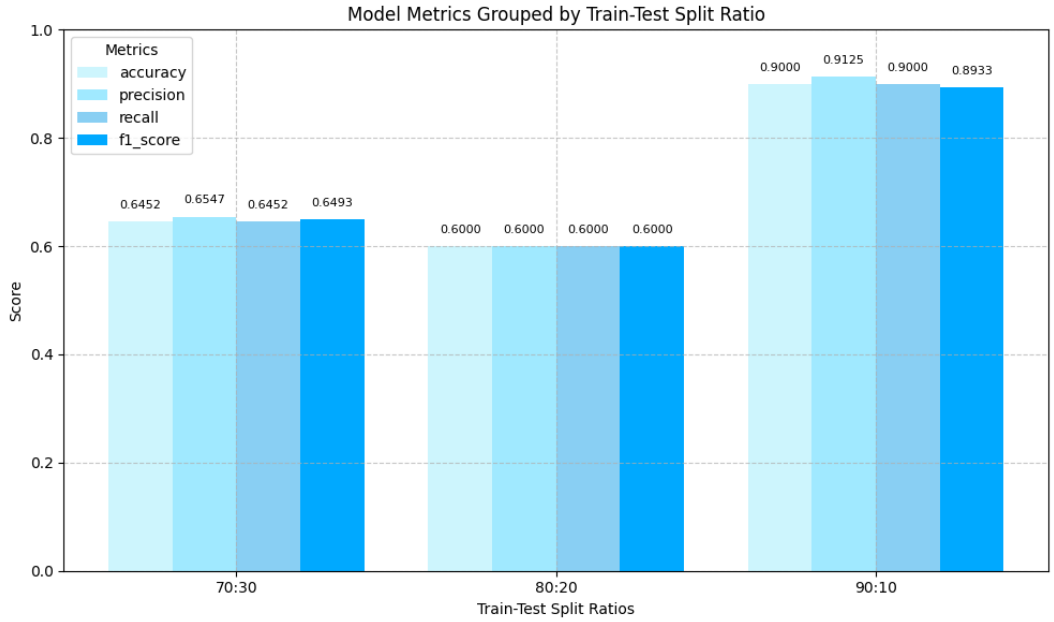
Uji Coba	Variasi Kategori	K-Fold Cross Validation	*Accuracy*	*Precision*	*Recall*	*F1-Score*
2	Produk Proses	k-10	72%	72%	72%	72%
		k-15	74%	74%	74%	74%
		k-20	71%	71%	71%	71%

Pengujian pada kategori *Produk* dan *Proses* menunjukkan performa model yang lebih beragam dibandingkan pengujian sebelumnya. Pendekatan K-Fold Cross Validation dengan variasi jumlah lipatan (k) menghasilkan pola yang menarik dalam performa model.

Pada pengujian dengan k-10, model menunjukkan performa dengan *Accuracy* dan *F1-Score* masing-masing sebesar 72%. Hasil mencerminkan bahwa model mempunyai kemampuan dasar yang cukup baik untuk mengklasifikasikan kategori produk dan proses, meskipun masih memerlukan peningkatan.

Ketika jumlah lipatan ditingkatkan menjadi k-15, performa model mengalami peningkatan pada semua metrik, yang masing-masing mencapai 74%. Peningkatan ini menunjukkan bahwa validasi dengan lebih banyak lipatan membantu memperbaiki kemampuan generalisasi model, sehingga menghasilkan prediksi yang lebih akurat.

Namun, pada k-20, performa model justru menurun dengan nilai *Accuracy* dan *F1-Score* kembali ke 71%. Penurunan ini mengindikasikan bahwa dengan jumlah lipatan yang lebih besar, model mungkin mengalami kesulitan dalam mempertahankan konsistensi performa. Hal ini juga dapat disebabkan oleh variasi dalam distribusi data antar-lipatan yang lebih kecil.



Gambar 4. 20 *Metrics Performa* Uji Coba Skenario 2

Berdasarkan gambar 4.20 di atas, hasil yang ditampilkan, model menunjukkan performa yang stabil di berbagai nilai k, dengan beberapa perubahan kecil pada nilai metrik ketika jumlah lipatan meningkat. Pada k-Fold dengan k-10, model mencapai skor 0.72 untuk semua metrik. Hal ini menunjukkan bahwa performa model cukup baik dan seimbang, di mana tidak ada satu metrik pun yang lebih rendah secara signifikan dibandingkan metrik lainnya. Performa ini mengindikasikan bahwa model dapat menghasilkan prediksi yang cukup konsisten.

Ketika jumlah lipatan meningkat menjadi k-15, terjadi sedikit peningkatan pada seluruh metrik, dengan nilai masing-masing naik menjadi 0.74. Hal ini menunjukkan bahwa dengan validasi silang menggunakan lebih banyak lipatan, model diuji pada variasi data yang lebih luas, sehingga menghasilkan evaluasi performa yang lebih optimal. Peningkatan ini menunjukkan bahwa model dapat memanfaatkan distribusi data yang lebih beragam untuk meningkatkan prediksinya.

Namun, pada k-Fold dengan k-20, performa model sedikit menurun dibandingkan dengan k-15, dengan nilai semua metrik menjadi 0.71. Penurunan ini dapat terjadi karena semakin banyaknya lipatan menyebabkan setiap subset data yang digunakan dalam pengujian menjadi lebih kecil, sehingga distribusi data dalam lipatan tersebut menjadi kurang representatif. Meskipun begitu, perbedaan ini tidak terlalu signifikan, yang memberi petunjuk bahwa model tetap mempunyai kemampuan dalam mengeneralisasi dengan baik.

Secara keseluruhan, performa terbaik model dicapai pada k-15, meskipun perbedaan dengan k-10 atau k-20 tidak terlalu besar. Stabilitas performa ini mengindikasikan bahwa model mampu menghadapi perubahan jumlah lipatan pada validasi silang tanpa kehilangan kualitas prediksinya. Oleh karena itu, penggunaan k-15 bisa menjadi opsi yang ideal untuk memperoleh hasil evaluasi yang sedikit lebih optimal.

Tabel 4. 4 Hasil Uji Coba Skenario 3

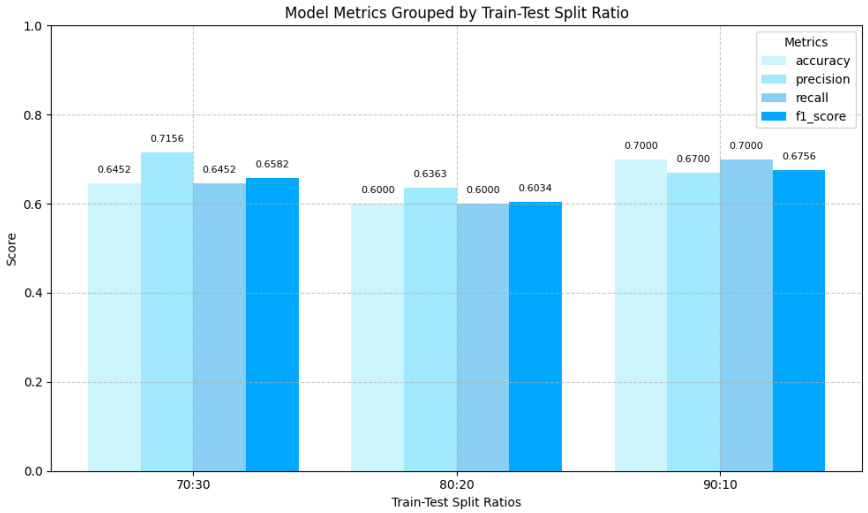
Uji Coba	Variasi Kategori	K-Fold Cross Validation	*Accuracy*	*Precision*	*Recall*	*F1-Score*
3	fungsional produk, fungsional proses, nonfungsional produk dan nonfungsional proses	k-10	61%	65%	61%	62%
		k-15	65%	65%	65%	64%
		k-20	64%	67%	64%	65%

Pada pengujian dengan kategori yang lebih kompleks, yaitu *Fungsional Produk*, *Fungsional Proses*, *Nonfungsional Produk*, dan *Nonfungsional Proses*, performa model menunjukkan variasi yang menarik ketika menggunakan pendekatan *K-Fold Cross Validation* mempunyai jumlah lipatan berbeda.

Pada k-10, model menghasilkan *Accuracy* sebesar 61%, dengan *F1-Score* 62%. Hasil ini menunjukkan bahwa meskipun tingkat ketepatan relatif lebih tinggi, model memiliki kesulitan dalam mempertahankan keseimbangan antara sensitivitas dan ketepatan, yang tercermin dalam nilai *F1-Score* yang lebih timggi.

Ketika jumlah lipatan ditingkatkan menjadi k-15, performa model meningkat dengan *Accuracy* mencapai 65%, serta *F1-Score* sedikit lebih rendah di angka 64%. Peningkatan ini menunjukkan bahwa validasi dengan lebih banyak lipatan membantu model menangkap pola dengan lebih baik, sehingga menghasilkan klasifikasi yang lebih konsisten.

Pada k-20, model menunjukkan performa yang stabil dengan sedikit peningkatan pada beberapa metrik. *Accuracy* berada di angka 64% dan *F1-Score* juga menunjukkan peningkatan tipis ke 65%. Hasil ini menandakan bahwa model semakin mampu menjaga keseimbangan antara ketepatan dan sensitivitas dengan bertambahnya jumlah lipatan.



Gambar 4. 21 *Metrics Performa* Uji Coba Skenario 3

Bedasarkan gambar 4.21 k-Fold dengan k-10, nilai metrik yang dicapai adalah 0.61 untuk accuracy dan 0.62 untuk F1-score. Hasil ini menunjukkan performa awal yang cukup seimbang.

Ketika jumlah lipatan ditingkatkan menjadi k-15, performa model menunjukkan sedikit peningkatan pada sebagian besar metrik. Accuracy naik ke 0.64, sedangkan F1-score tetap stabil di 0.65. Peningkatan ini mengindikasikan bahwa pengujian dengan lebih banyak lipatan membantu model memanfaatkan distribusi data yang lebih luas, sehingga evaluasi performanya menjadi lebih akurat.

Pada k-Fold dengan k-20, performa model menunjukkan stabilitas dengan metrik accuracy dan F1-score berada di 0.64. Niali metrik yang lebih tinggi pada k ini mengindikasikan bahwa model semakin baik dalam menghindari prediksi positif palsu, meskipun metrik lainnya tetap stabil.

Secara keseluruhan, performa model menunjukkan stabilitas di semua nilai k, dengan peningkatan yang konsisten dari k-10 ke k-15 dan sedikit peningkatan pada k-20. Perbedaan performa antar nilai k tidak terlalu signifikan, yang memberi petunjuk bahwa model mempunyai kemampuan generalisasi dengan baik. Untuk evaluasi yang optimal, penggunaan k-15 atau k-20 dapat dipertimbangkan, dengan k-20 memberikan hasil precision terbaik.

Dalam penelitian yang telah dilakukan, model dapat digunakan untuk melakukan klasfikasi terharap *software requirement.* Model ini diharapkan dapat mengatasi inkonsistensi dalam mengidentifikasi persyaratan perangkat lunak yang tidak memadai. Dalam hal ini, berkaitan dengan surat ayat ini,

لَا يُكَلِّفُ ٱللَّهُ نَفْسًا إِلَّا وُسْعَهَا

"Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya..." (QS Al-Baqarah:256)

Tafsir Ibnu Katsir menjelaskan bahwa QS. Al-Baqarah ayat 286 menunjukkan betapa Allah Maha Bijaksana dan Maha Pengasih, yang tidak membebani manusia melebihi kemampuan mereka (*Surat Al-Baqarah Ayat 286,* 2023.). Prinsip tidak membebani melainkan sesuai dengan kesanggupannya dapat diimplementasikan dengan memastikan bahwa kebutuhan perangkat lunak dikelompokkan secara efisien dan sesuai konteks pengguna. Pengelompokan kebutuhan dengan algoritme yang tepat, seperti Naïve Bayes, membantu pengembang meminimalkan beban kerja manual dan meningkatkan akurasi pengelolaan data.

Dalam mengklasifikasikan kebutuhan, prinsip keadilan dalam QS. Al-Baqarah:286 diterapkan dengan memastikan bahwa semua kebutuhan pengguna diperlakukan sama pentingnya sesuai dengan kategori dan prioritasnya. Misalnya, kebutuhan kecil tidak diabaikan, tetapi dimasukkan ke dalam kategori yang tepat untuk diakomodasi. Data kebutuhan harus diklasifikasikan berdasarkan fakta dan analisis yang obyektif. Hasil prediksi dari Naïve Bayes juga harus digunakan secara transparan dalam pengambilan keputusan pengembangan perangkat lunak.

Dengan menerapkan algoritme Naïve Bayes dalam klasifikasi *software requirement*, prinsip QS. Al-Baqarah:286 dapat diintegrasikan melalui pengelolaan yang adil, efisien, dan memperhatikan batas kemampuan manusia. Hal ini tidak hanya mempermudah proses pengembangan perangkat lunak tetapi juga

mencerminkan nilai-nilai Islami yang mendukung keseimbangan, tanggung jawab, dan keadilan dalam setiap aspek pekerjaan.

الَّذِيْنَ يَسْتَمِعُوْنَ الْقَوْلَ فَيَتَّبِعُوْنَ اَحْسَنَهٗۗ اُولٰۤىِٕكَ الَّذِيْنَ هَدٰىهُمُ اللّٰهُ وَاُولٰۤىِٕكَ هُمْ اُولُوا الْاَلْبَابِ

"(yaitu) mereka yang mendengarkan perkataan lalu mengikuti apa yang paling baik di antaranya. Mereka itulah orang-orang yang telah diberi petunjuk oleh Allah dan mereka itulah orang-orang yang mempunyai akal sehat." (QS. Az-Zumar 39: Ayat 18)

Tafsir Wajiz menjelaskan bahwa Ayat Surah Az-Zumar (39:18) berbicara yaitu orang-orang yang mendengarkan ajaran, baik dari Al-Qur'an maupun hadis, kemudian memilih dan mengikuti yang terbaik di antaranya, karena wahyu Allah adalah yang paling sempurna. Mereka inilah yang telah diberikan petunjuk oleh Allah, memiliki pemikiran jernih, dan tidak terpengaruh oleh kebingungan atau keraguan (*Surat Az-Zumar Ayat 18*, 2024). Ayat ini mengajarkan pentingnya mendengar berbagai pandangan atau pendapat, lalu memilih yang paling baik di antaranya. Dalam konteks pengelompokan kebutuhan (*software requirement classification*), prinsip ini bisa diterapkan secara lebih rinci dalam beberapa cara, terutama terkait bagaimana kita mendengarkan dan mengelompokkan kebutuhan yang diungkapkan oleh berbagai pemangku kepentingan.

اِنَّا كُلَّ شَيْءٍ خَلَقْنٰهُ بِقَدَرٍ

"Sesungguhnya Kami menciptakan segala sesuatu menurut ukuran." (QS. Al-Qamar 54: Ayat 49)

Berdasarkan Tafsir Wajiz pada Surah Al-Qamar (54:49) berisi segala sesuatu yang terjadi pada makhluk telah ditetapkan oleh Allah. Sesungguhnya, Kami menciptakan segala sesuatu sesuai dengan ukuran, yakni dalam sistem dan ketentuan

yang telah ditentukan sebelumnya (*Surat Al-Qamar Ayat 49*, 2024). Dalam konteks klasifikasi *software requirement* dapat dihubungkan melalui beberapa prinsip yang diterapkan dalam pengelompokan dan pengelolaan kebutuhan perangkat lunak. Ayat ini memberi petunjuk bahwa segala hal yang ada di alam semesta mempunyai ukuran, aturan, dan keseimbangan yang telah Allah tetapkan. Prinsip-prinsip ini sangat relevan untuk memahami bagaimana kebutuhan perangkat lunak dapat diklasifikasikan dengan tepat sesuai dengan tujuannya. Dalam pengembangan perangkat lunak, ada berbagai jenis kebutuhan, seperti kebutuhan fungsional, kebutuhan non-fungsional, produk dan proses. Prinsip dari ayat ini mengajarkan bahwa setiap klasifikasi harus dilakukan dengan cermat dan tepat, sesuai dengan prioritas yang sudah ditetapkan. Kebutuhan yang lebih kritis, misalnya, harus mendapat perhatian lebih besar dibandingkan yang bersifat opsional atau tambahan. Ini mencerminkan prinsip takaran atau ukuran yang ditetapkan oleh Allah dalam ciptaan-Nya.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Masalah utama dalam pengelolaan persyaratan perangkat lunak adalah inkonsistensi dalam mengidentifikasi persyaratan, yang sering menyebabkan kebingungan, kesalahan pemahaman, dan dampak negatif pada proses pengembangan perangkat lunak. Penanganan persyaratan yang tidak memadai juga dapat memicu hilangnya kebutuhan kritis, sehingga mempengaruhi kualitas perangkat lunak secara keseluruhan.

Dalam mengatasi masalah ini, klasifikasi persyaratan perangkat lunak menggunakan metode seperti *Multinomial Naïve Bayes* dapat menjadi solusi yang efektif. Dengan mengelompokkan persyaratan berdasarkan kategori seperti fungsional dan nonfungsional, produk dan proses, atau kombinasi keduanya yaitu fungsional produk, fungsional proses, nonfungsional produk dan nonfungsional proses., pengembang dapat mengidentifikasi kebutuhan secara lebih akurat, memastikan prioritas yang tepat, dan meningkatkan keandalan serta efisiensi proses pengembangan. Proses pengujian skenario menggunakan *k-fold cross validation* yaitu k-10, k15, dan k-20.

Dengan mengklasifikasikan persyaratan perangkat lunak berdasarkan kategori menandakan bahwa masalah yang dihadapi berhasil diatasi. Penggunaan metode *Naïve Bayes* memberikan hasil yang baik dalam pengklasifikasian yang telah dilakukan. Penggunaan *K-Fold* pada skenario pertama dengan kategori

fungsional dan nonfungsional mengasilkan kesimpulan berupa model mencapai performa terbaiknya pada k-15 dan k-20 dengan skor 0.85, meskipun perbedaannya dengan k-10 tidak terlalu signifikan sebesar 0.84. Pada skenario kedua dengan kategori produk dan proses mencapai kesimpulan berupa model menunjukkan performa yang stabil pada berbagai nilai k dengan performa terbaik dicapai pada k-15, di mana semua metrik mencapai skor 0.74. Sementara perbedaan performa dengan k-10 atau k-20 tidak signifikan dengan metrik mencapai skor masing-masing 0.72 dan 0.71. Skenario ketiga yang terakhir dengan kategori fungsional produk, fungsional proses, nonfungsional produk dan nonfungsional proses mencapai kesimpulan berupa model menunjukkan performa yang stabil di berbagai nilai k, dengan peningkatan konsisten dari k-10 ke k-15 dan precision terbaik pada k-20 (0.67). Meskipun perbedaan antar nilai k tidak signifikan, k-15 atau k-20 dapat dipilih untuk evaluasi optimal, dengan k-20 unggul dalam precision. Stabilitas hasil yang diperoleh pada skenario pertama, kedua dan ketiga mencerminkan kemampuan generalisasi model yang baik.

5.2 Saran

Berdasarkan proses dan hasil penelitian ini, peneliti menyadari bahwa masih terdapat banyak kekurangan dalam penelitian ini. Maka dari itu, diharapkan pada penelitian selanjutnya Untuk dapat melakukan perbaikan dan peningkatan agar hasil yang diperoleh lebih baik, berikut beberapa saran untuk penelitian selanjutnya:

1. Menggunakan dataset yang lebih banyak dan beragam. Hal ini akan membantu mengukur kemampuan model dalam menghadapi variasi data yang lebih luas.

2. Penelitian selanjutnya dapat mencoba membandingkan denga metode lain yang berbeda.
3. Menambahkan fitur tambahan selain menggunakan TF-IDF dapat membantu menangkap informasi yang lebih kompleks dalam mengklasifikan teks.

DAFTAR PUSTAKA

Abdullahi, S., Ahmed Zayyad, M., Yusuf, N., Bagiwa, L. I., Nura, A., Zakari, A., & Dansambo, B. (2021). Software Requirements Negotiation: A Review on Challenges. *International Journal of Innovative Computing*, *11*(1), 1–6. https://doi.org/10.11113/ijic.v11n1.264

Alshazly, A. A., Elfatatry, A. M., & Abougabal, M. S. (2014). Detecting defects in software requirements specification. *Alexandria Engineering Journal*, *53*(3), 513–527. https://doi.org/10.1016/j.aej.2014.06.001

Baker, C., Deng, L., Chakraborty, S., & Dehlinger, J. (2019). Automatic multi-class non-functional software requirements classification using neural networks. *Proceedings - International Computer Software and Applications Conference*, *2*, 610–615. https://doi.org/10.1109/COMPSAC.2019.10275

Buchori, A., Khotijah, S., & Ramdan, A. S. (2022). Sistem Pakar Diagnosa Penyakit Paru-Paru Menggunakan Metode Naive Bayes Classifier Berbasis Java *Jurnal Semnas Ristek (Seminar Nasional Riset dan Inovasi Teknologi) 5645-10629-1-Sm*. 1–7. https://doi.org/10.30998/semnasristek.v6i1.5645

Devita, R. N., Herwanto, H. W., & Wibawa, A. P. (2018). Perbandingan Kinerja Metode Naive Bayes dan K-Nearest Neighbor untuk Klasifikasi Artikel Berbahasa indonesia. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *5*(4), 427–434. https://doi.org/10.25126/jtiik.201854773

Hartijo, B., Aini, K. N., & Murtiyasa, B. (2018). Klasifikasi Dokumen berkonten Serangan jaringan menggunakan Multinomial Naive Bayes. *Seminar Nasional Teknologi Informasi Dan Komunikasi (SEMNASTIK)*, *1*(1), 112–118.https://www.academia.edu/106320857/Klasifikasi_Dokumen_berkonten_Serangan_jaringan_menggunakan_Multinomial_Naive_Bayes

Martins, H. F., de Oliveira, A. C., Canedo, E. D., Kosloski, R. A. D., Paldês, R. Á., & Oliveira, E. C. (2019). Design thinking: Challenges for software requirements elicitation. *Information (Switzerland)*, *10*(12), 1–27. https://doi.org/10.3390/info10120371

Mayasari, L., & Indarti, D. (2022). Klasifikasi Topik Tweet Mengenai Covid Menggunakan Metode Multinomial Naïve Bayes Dengan Pembobotan Tf-Idf. *Jurnal Ilmiah Informatika Komputer*, *27*(1), 43–53. https://doi.org/10.35760/ik.2022.v27i1.6184

Nita Yunitasari, N. Y., Fitri, S., & Taufiq, M. (2022). Perancangan Media Pembelajaran Berbasis Android Pada Materi Trigonometri Untuk Peserta Didik Kelas X di SMA Negeri 1 Cipatujah. *Produktif : Jurnal Ilmiah Pendidikan Teknologi Informasi*, *5*(1), 419–426. https://doi.org/10.35568/produktif.v5i1.1007

Rahimi, N., Eassa, F., & Elrefaei, L. (2020). An ensemble machine learning technique for functional requirement classification. *Symmetry*, *12*(10), 1–26. https://doi.org/10.3390/sym12101601

Ramadandi, S., & Jahring, J. (2020). Klasifikasi Gaya Belajar Mahasiswa Menggunakan Metode Naïve Bayes Classifier. *Jurnal Teknologi Dan Informasi*, *10*(2), 170–179. https://doi.org/10.34010/jati.v10i2

Serafintino, R. A., & Susilowati, M. (2022). Dokumen Software Requirement Spesification (Srs) Sistem Informasi Pemasaran Usaha Jasa Percetakan Dan Iklan. *Kurawal - Jurnal Teknologi, Informasi Dan Industri*, *5*(2), 117–128. https://doi.org/10.33479/kurawal.v5i2.646

Bourque, Pierre and Fairley, R.E.. (2014). *Guide to the Software Engineering Body of Knowledge Version 3.0 (SWEBOK Guide V3.0)*. https://ieeecs-media.computer.org/media/education/swebok/swebok-v3.pdf

Sulastri, S., & Zuliardo, E. (2020). Klasifikasi Dokumen Dengan Metode Naive Bayes Terhadap Putusan Kasasi Pengadilan Tentang Merk. *Jurnal Dinamika Informatika*, *12*(1), 45–52. https://doi.org/10.35315/informatika.v12i1.8162

Surat Al-Baqarah Ayat 286 Arab, Latin, Terjemah dan Tafsir | Baca di TafsirWeb. (2023). Retrieved November 30, 2024, from https://tafsirweb.com/1052-surat-al-baqarah-ayat-286.html

Surat Al-Qamar Ayat 49: Arab, Latin, Terjemah dan Tafsir Lengkap | Quran NU Online. (2024). Retrieved November 30, 2024, from https://quran.nu.or.id/al-qamar/49

Surat Az-Zumar Ayat 18: Arab, Latin, Terjemah dan Tafsir Lengkap | Quran NU Online. (2024). Retrieved November 30, 2024, from https://quran.nu.or.id/az-zumar/18

Tiun, S., Mokhtar, U. A., Bakar, S. H., & Saad, S. (2020). Classification of functional and non-functional requirement in software requirement using Word2vec and fast Text. *Journal of Physics: Conference Series*, *1529*(4). https://doi.org/10.1088/1742-6596/1529/4/042077

Vineetha, V. K., & Samuel, P. (2022). A Multinomial Naïve Bayes Classifier for identifying Actors and Use Cases from Software Requirement Specification documents. *2022 2nd International Conference on Intelligent Technologies, CONIT 2022*, 1–5. https://doi.org/10.1109/CONIT55038.2022.9848290

Wati, R., Ernawati, S., & Rachmi, H. (2023). Pembobotan TF-IDF Menggunakan Naïve Bayes pada Sentimen Masyarakat Mengenai Isu Kenaikan BIPIH. *Jurnal Manajemen Informatika (JAMIKA)*, *13*(1), 84–93. https://doi.org/10.34010/jamika.v13i1.9424

Yaseen, M., Mustapha, A., Ali, Z., Zaman, S. U., & Kamal, S. W. (2020). Using Cumulative Voting and Priority Groups To Prioritize Functional Requirements: Odoo Erp As Case Study. *Journal of Software Engineering & Intelligent Systems*, *5*(2), 1–9. https://doi.org/10.26634/jse.14.4.17726

LAMPIRAN