RANCANG BANGUN CHATBOT "GUMARA" SEBAGAI *VIRTUAL ROUTE GUIDE* WISATAWAN ASING MENGGUNAKAN METODE *RANDOM FOREST CLASSIFIER*

SKRIPSI

Oleh :
ALFIN RIZKY AMARTYA
NIM. 19650065

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2025

RANCANG BANGUN CHATBOT "GUMARA" SEBAGAI *VIRTUAL ROUTE GUIDE* WISATAWAN ASING MENGGUNAKAN METODE *RANDOM FOREST CLASSIFIER*

SKRIPSI

Diajukan Kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk mmemenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
ALFIN RIZKY AMARTYA
NIM. 19650065

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2025

HALAMAN PERSETUJUAN

RANCANG BANGUN CHATBOT "GUMARA" SEBAGAI *VIRTUAL ROUTE GUIDE* WISATAWAN ASING MENGGUNAKAN METODE *RANDOM FOREST CLASSIFIER*

SKRIPSI

Oleh :
ALFIN RIZKY AMARTYA
NIM. 19650065

Telah diperiksa dan disetujui untuk diuji
Tanggal : 15 Desember 2023

Pembimbing I	Pembimbing II
Puspa Miladin N.S.A.B., M.Kom	Dr. Ir. Fresy Nugroho, ST., MT, IPM.
NIP. 19930828 201903 2 018	NIP. 19710722 201101 1 001

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

Supriyono, M. Kom
NIP. 19841010 201903 1 012

HALAMAN PENGESAHAN

RANCANG BANGUN CHATBOT "GUMARA" SEBAGAI *VIRTUAL ROUTE GUIDE* WISATAWAN ASING MENGGUNAKAN METODE *RANDOM FOREST CLASSIFIER*

SKRIPSI

Oleh :
ALFIN RIZKY AMARTYA
NIM. 19650065

Telah dipertahankan di depan Dewan Penguji
dan dinyatakan diterima sebagai salah satu persyaratan
untuk memperoleh gelar Sarjana Komputer (S.Kom)
Pada Tanggal: 15 Desember 2023

Susunan Dewan Penguji

Ketua Penguji	:	Hani Nurhayati, M.T NIP. 19780625 200801 2 006	( )
Anggota Penguji I	:	Dr. Agung Teguh Wibowo Almais, M.T NIP. 19860103201802011235	( )
Anggota Penguji II	:	Puspa Miladin N.S.A.B., M.Kom NIP. 19930828 201903 2 018	( )
Anggota Penguji III	:	Dr. Ir. Fresy Nugroho, ST., MT, IPM NIP. 19710722 201101 1 001	( )

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

Supriyono, M. Kom
NIP. 19841010 201903 1 012

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	:	Alfin Rizky Amartya
NIM	:	19650065
Fakultas / Program Studi	:	Sains dan Teknologi / Teknik Informatika
Judul Skripsi	:	Rancang Bangun *Chatbot* "Gumara" Sebagai *Virtual Route Guide* Wisatawan Asing Menggunakan Metode *Random Forest Classifier*

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

Malang, 15 Desember 2023
Yang membuat pernyataan,

Alfin Rizky Amartya
NIM. 19650065

MOTTO

"*TOTO, TATAG, TUTUG*"

(H. Djimat Hendro Soewarno)

HALAMAN PERSEMBAHAN

Sebagai tanda bakti, hormat, dan terimakasih yang tak terhingga, saya persembahkan karya kecil ini kepada:

1. Keluarga tercinta Bapak Zainal Arifin dan Ibu Nunuk Sulistiani selaku orang tua penulis, serta Devin Yusuf Faturrahman selaku adik penulis atas segala dukungan dalam segala hal serta kepercayaan kepada penulis untuk menyelesaikan skripsi ini dalam waktu yang tidak biasa.
2. Puspa Miladin Nuraida Safitri A Basid, M.Kom selaku dosen pembimbing I serta Dr. Ir. Fresy Nugroho, ST., MT, IPM. selaku dosen pembimbing II yang telah bersedia meluangkan waktunya dalam membimbing dan memberi arah kepada penulis sehingga dapat menyelesaikan skripsi ini.

KATA PENGANTAR

Segala puji dan syukur penulis limpahkan atas kehadirat Allah SWT yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis mampu menyelesaikan skripsi ini dengan baik. Sholawat serta salam tetap tercurahkan kepada junjungan kita Nabi Muhammad SAW atas syafaatnya yang telah menuntun umat islam menuju jalan yang baik. Semoga kita semua termasuk dalam golongan yang dituntun Allah SWT dan mendapat pertolongan Nabi Muhammad SAW. Aamiin.

Penulis sangat menyadari bahwa penulis masih sangat minim ilmu dan pengetahuan, sehingga tanpa adanya peran dan kontribusi dari pihak yang telah membantu meluangkan waktu dan memberikan sumbangsih pemikiran dalam membimbing penulis, penulis tidak akan mampu menyelesaikan skripsi ini dengan baik. Pada bagian ini juga segala kerendahan hati, penulis menyampaikan terima kasih kepada :

1. Prof. Dr. Hj. Ilfi Nur Diana, M.Si., selaku Rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Dr. Agus Mulyono, M.Kes., selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Supriyono, M. Kom, selaku Ketua Program Studi Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Puspa Miladin Nuraida Safitri A Basid, M.Kom selaku dosen pembimbing I dan Dr. Ir. Fresy Nugroho, ST., MT, IPM selaku dosen pembimbing II atas bimbingan, arahan, dan motivasi yang sangat berarti bagi penulis.

5. Hani Nurhayati, M.T, selaku Dosen Penguji I sekaligus wali dosen penulis selama kuliah dan Dr. Agung Teguh Wibowo Almais, M.T selaku Dosen Penguji II yang telah meluangkan waktu memberikan arahan untuk skripsi ini.
6. Seluruh Dosen dan Staf di Program Studi Teknik Informatika, dengan ikhlas memberikan ilmu, bantuan, serta dorongan semangat selama perkuliahan.
7. Keluarga yang selalu memberikan support saya terutama untuk Bapak Zainal Arifin, Ibu Nunuk Sulistiani, dan adik saya Devin Yusuf Faturrahman yang selalu memberikan segala dukungan kepada penulis.
8. Teruntuk teman-teman saya dan seseorang yang menemani dan membantu penulis, memberi dukungan penuh, serta memotivasi dalam mengerjakan sehingga skripsi ini dapat terselesaikan terutama Adinda Putri Aprilianti.
9. Saudara-saudara seasuhan Persaudaraan Setia-Hati Winongo Tunas Muda Madiun khususnya Cabang Kabupaten Malang yang telah memotivasi dan memberikan dukungan untuk menyelesaikan skripsi ini
10. Seluruh pihak yang telah terlibat secara langsung maupun tidak langsung dalam membantu menyelesaikan skripsi ini.

Penulis sangat menyadari bahwa dalam penyusunan skripsi ini masih terdapat kekurangan dan penulis berharap skripsi ini dapat memberikan manfaat bagi pembaca maupun bagi penulis.

Malang, 15 Desember 2023

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Amartya, Alfin Rizky. 2025. **"Rancang Bangun Chatbot Gumara Sebagai *Virtual Route Guide* Wisatawan Asing Menggunakan Metode *Random Forest Classifier"*.** Skripsi. Jurusan Teknik Informatika, Fakultas Sains dan Teknologi. Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Puspa Miladin Nuraida Safitri A Basid, M.Kom, (II) Dr. Ir. Fresy Nugroho, ST., MT, IPM.

Kata Kunci : *Random Forest Classifier*, *Virtual Route Guide*, Pariwisata, Chatbot

Pariwisata telah menjadi industri yang krusial dalam perekonomian global. Potensi pariwisata dalam memberikan pemasukan bagi daerah yang menggarapnya dengan baik telah menjadi fokus utama, salah satunya di Malang Raya, sebuah kawasan yang kaya akan destinasi wisata. Namun, semakin banyaknya destinasi wisata di Malang Raya juga menghadirkan tantangan baru, terutama bagi wisatawan mancanegara dalam mendapatkan informasi terkait rute, arah, dan jalan menuju destinasi yang diinginkan. Terdapat kendala dalam akses informasi yang memadai dari berbagai media yang ada, yang belum sepenuhnya memenuhi kebutuhan informasi wisatawan. Pada penelitian ini mengambil kasus tentang urgensi sulitnya wisatawan mancanegara dalam mendapatkan informasi terkait arah, rute, maupun jalan akses menuju ke destinasi wisata yang dituju khususnya di wilayah Malang Raya dikarenakan faktor informasi yang terbatas dari badan pemerintah yang menangani sektor pariwisata hingga faktor sulitnya komunikasi kepada warga lokal yang tidak semuanya bisa mengerti apa yang dikomunikasikan oleh para wisatawan mancanegara. Pada penelitian ini, diusulkan penggunaan chatbot interaktif sebagai *Virtual Route Guide* menggunakan metode *Random Forest Classifier*. Tujuan dari penelitian ini adalah mengukur hasil akurasi dari penggunaan metode klasifikasi *Random Forest Classifier*. Pada penelitian ini diperlukan inisialisasi daftar kata kunci dan aturan untuk fitur-fitur yang digunakan dalam proses klasifikasi sehingga bisa mendapatkan hasil performa lebih baik. Pada chatbot interaktif *Virtual Route Guide* menggunakan *Random Forest Classifier* didapatkan hasil terbaik dengan pembagian dataset 70:30 menggunakan kombinasi *hyperparameter* jumlah *tree* (*n_estimator*) 200, kedalaman *maximum tree* (*max_depth*) 20, dan Jumlah minimum sampel untuk membagi node (*min_sample_split*) 10 menghasilkan nilai akurasi sebesar 80,28%. Sehingga dapat disimpulkan bahwa pentingnya pengaturan *hyperparameter* yang tepat dan pemilihan dataset yang seimbang dalam mencapai hasil yang optimal pada klasifikasi dengan menggunakan metode *Random Forest Classifier* untuk mengembangkan chatbot sebagai *Virtual Route Guide*.

ABSTRACT

Amartya, Alfin Rizky. 2025. **"Design and Development of the Gumara Chatbot as a Virtual Route Guide for Foreign Tourists Using the Random Forest Classifier Method".** Thesis. Department of Informatics Engineering, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University, Malang. Supervisor: (I) Puspa Miladin Nuraida Safitri A Basid, M.Kom, (II) Dr. Fresy Nugroho, M. T

Keywords: *Random Forest Classifier*, Virtual Route Guide, Tourism, Chatbot

Tourism has become a crucial industry in the global economy. The potential of tourism to provide income for regions that develop it well has become a primary focus, one of which is in Malang Raya, an area rich in tourist destinations. However, the increasing number of tourist destinations in Malang Raya also presents new challenges, especially for foreign tourists in obtaining information regarding routes, directions, and roads to their desired destinations. There are constraints in accessing adequate information from various available media, which have not fully met the needs of tourists. This research focuses on the difficulty faced by foreign tourists in obtaining information about directions, routes, and access roads to tourist destinations, particularly in the Malang Raya region, due to limited information from government bodies handling the tourism sector and the difficulty in communicating with local residents, not all of whom can understand what is being communicated by foreign tourists. This study proposes the use of an interactive chatbot as a Virtual Route Guide using the Random Forest Classifier method. The aim of this research is to measure the accuracy of using the Random Forest Classifier classification method. This study requires the initialization of a list of keywords and rules for the features used in the classification process to achieve better performance results. The interactive chatbot Virtual Route Guide using the Random Forest Classifier obtained the best results with a dataset split of 70:30 using a combination of hyperparameters: number of trees (n_estimators) at 200, maximum tree depth (max_depth) at 20, and minimum samples required to split a node (min_sample_split) at 10, resulting in an accuracy value of 80.28%. Hence, it can be concluded that the proper adjustment of hyperparameters and the selection of a balanced dataset are crucial in achieving optimal results in classification using the Random Forest Classifier method to develop a chatbot as a Virtual Route Guide.

مستخلص البحث

أمارتيا ، ألفين رزقي .٢٠٢٥ .''تصميم جومارا شات بوت كدليل طريق افتراض للسياح الأجانب باستخدام طريقة تصنيف الغابات العشوائية'' .اطروحه .قسم الهندسة المعلوماتية، كلية العلوم والتكنولوجيا .جامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج

الكلمات الدالة: مصنف الغابات العشوائي ، دليل الطريق الافتراضي ، السياحة ، شات بوت

أصبحت السياحة صناعة حاسمة في الاقتصاد العالمي .أصبحت إمكانية السياحة في توفير الدخل للمناطقالتيتديرها بشكل جيد هي محور التركيز الرئيس. أحدها في مالانج الكبرى، وهي منطقة غنية بالوجهات السياحية. ومع ذلك، فإن العدد المتزايد من الوجهات السياحية في مالانج رايا يمثل أيضًا تحديات جديدة، خاصة بالنسبة للسياح الأجانب في الحصول على معلومات بشأن الطرق ،والاتجاهات والطرق المؤدية إلى وجهتهم المرغوبة .هناك عقبات في الوصول إلى المعلومات الكافية من مختلف وسائل الإعلام الموجودة والتي لا تلبي بشكل كامل احتياجات السياح من المعلومات .يأخذ هذا البحث حالة الضرورة الملحة للصعوبة التي يواجهها السياح الأجانب في الحصول على معلومات بشأن الاتجاهات والطرق وطرق الوصول إلى الوجهات السياحية المقصودة، خاصة في منطقة مالانج الكبرى بسبب محدودية المعلومات من الوكالات الحكومية التي تتعامل مع قطاع السياحة و صعوبة التواصل مع السكان، ولا يستطيع جميع السكان المحليين فهم ما يتم التواصل معه من قبل السياح الأجانب .في هذا البحث، يقترح استخدام روبوت الدردشة التفاعلي كدليل طريق افتراضي باستخدام طريقة باستخدام طريقة تصنيف الغابات العشوائية .الهدف من هذا البحث هو قياس دقة نتائج استخدام طريقة التصنيف باستخدام طريقة تصنيف الغابات العشوائية .من الضروري في هذا البحث تهيئة قائمة من الكلمات الرئيسية والقواعد الخاصة بالميزات المستخدمة في عملية التصنيف حتى يمكن الحصول على نتائج أداء أفضل .في برنامج الدردشة التفاعلي طريق افتراضي للسياح باستخدام باستخدام طريقة تصنيف الغابات العشوائية ، تم الحصول على أفضل النتائج عن طريق ٢٠٠ والحد الأقصى (*n_estimator*) تقسيم مجموعة البيانات ٧٠:٣٠ باستخدام مجموعة من المعلمات الفائقة لعدد الأشجار ٥ لإنتاج (*min_sample_split*) ٢٠ والحد الأدنى لعدد الأشجار عينات لتقسيم العقد (*max_depth*) لعمق الشجرة دقة قيمة تبلغ ٩٥،٨٨٪ لذلك يمكن الاستنتاج أن أهمية تحديد المعلمات الفائقة الصحيحة واختيار مجموعة بيانات متوازنة في تحقيق . النتائج المثلى في التصنيف باستخدام طريقة باستخدام طريقة تصنيف .الغابات العشوائية لتطوير روبوت الدردشة كدليل طريق افتراضي

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pariwisata merupakan kegiatan yang dinamis, melibatkan banyak individu, dan menghidupkan berbagai sektor usaha. Di era globalisasi sekarang, pariwisata dianggap sebagai penggerak utama ekonomi dunia dan menjadi industri yang universal. Pariwisata dapat memberikan pendapatan signifikan bagi daerah yang memahami potensinya dalam sektor pariwisata (Ismayanti, 2010).

Fungsi pariwisata juga terkait dengan dimensi spiritual, yaitu memperkuat iman. Dengan mengamati kebesaran Allah melalui langit, bumi, dan pergantian siang dan malam, diharapkan seorang muslim akan lebih menyadari bahwa dirinya adalah ciptaan Allah SWT dan rezekinya berasal dari-Nya. Hal ini sesuai dengan firman Allah dalam Al-Qur'an, surat Al-An'am. ayat 11-12.

قُلْ سِيْرُوْا فِى الْاَرْضِ ثُمَّ انْظُرُوْا كَيْفَ كَانَ عَاقِبَةُ الْمُكَذِّبِيْنَ ١١ قُلْ لِّمَنْ مَّا فِى السَّمٰوٰتِ وَالْاَرْضِۗ قُلْ لِّلّٰهِۗ كَتَبَ عَلٰى نَفْسِهِ الرَّحْمَةَۗ لَيَجْمَعَنَّكُمْ اِلٰى يَوْمِ الْقِيٰمَةِ لَا رَيْبَ فِيْهِۗ اَلَّذِيْنَ خَسِرُوْٓا اَنْفُسَهُمْ فَهُمْ لَا يُؤْمِنُوْنَ ١٢

"Katakanlah (Nabi Muhammad), Jelajahilah bumi, kemudian perhatikanlah bagaimana kesudahan orang-orang yang mendustakan itu; (11) Katakanlah (Nabi Muhammad), "Milik siapakah apa yang di langit dan di bumi?" Katakanlah, "Milik Allah." Dia telah menetapkan (sifat) kasih sayang pada diri-Nya. Sungguh, Dia pasti akan mengumpulkan kamu pada hari Kiamat yang tidak ada keraguan padanya. Orang-orang yang merugikan dirinya, mereka itu tidak beriman. (12)"
(Q.S. Al-An'am : 11-12)

Menurut tafsir Ibnu Katsir jilid 5, Allah menyuruh Nabi Muhammad agar mengajak kaumnya untuk mengembara di atas bumi, mengunjungi tempat-tempat bersejarah di mana bangsa-bangsa terdahulu yang memusuhi rasul-rasul

dibinasakan. Tujuannya adalah agar mereka merenung mengapa bangsa-bangsa kuat tersebut hancur. Meskipun kaum Mekkah sudah sering melakukan perjalanan dagang, mereka diingatkan untuk memperhatikan bekas peninggalan bangsa-bangsa yang musnah. Allah menjelaskan bahwa banyak generasi yang telah diganti setelah dibinasakan. Hal ini seharusnya menjadi pelajaran sejarah bagi mereka. Ayat ini memberikan hiburan kepada Nabi Muhammad karena mengisyaratkan kekalahan bagi kaum musyrik.

Di sisi lain, pariwisata juga memiliki peran yang terkait dengan dimensi spiritual, yaitu meningkatkan kegiatan dzikir dan tafakkur. Saat seorang muslim mengamati kebesaran Allah melalui langit, bumi, dan pergantian siang dan malam, hal ini akan meningkatkan kesadaran tafakkur dalam dirinya. Sesuai dengan firman Allah SWT dalam surat Al-'Imran ayat 190-191:

اِنَّ فِيْ خَلْقِ السَّمٰوٰتِ وَالْاَرْضِ وَاخْتِلَافِ الَّيْلِ وَالنَّهَارِ لَاٰيٰتٍ لِّاُولِى الْاَلْبَابِۙ ١٩٠ الَّذِيْنَ يَذْكُرُوْنَ اللّٰهَ قِيَامًا وَّقُعُوْدًا وَّعَلٰى جُنُوْبِهِمْ وَيَتَفَكَّرُوْنَ فِيْ خَلْقِ السَّمٰوٰتِ وَالْاَرْضِۚ رَبَّنَا مَا خَلَقْتَ هٰذَا بَاطِلًاۚ سُبْحٰنَكَ فَقِنَا عَذَابَ النَّارِ ١٩١

"Sesungguhnya dalam penciptaan langit dan bumi serta pergantian malam dan siang terdapat tanda-tanda (kebesaran Allah) bagi orang yang berakal; (190) Yaitu orang-orang yang mengingat Allah sambil berdiri, duduk, atau dalam keadaan berbaring, dan memikirkan tentang penciptaan langit dan bumi (seraya berkata), "Ya Tuhan kami, tidaklah Engkau menciptakan semua ini sia-sia. Mahasuci Engkau. Lindungilah kami dari azab neraka. (191)" (Q.S. Al-'Imran : 190-191)

Dalam tafsir jalalain, Nabi menyatakan bahwa celakalah bagi yang membacanya tanpa merenungkan artinya. Merenungkan pergantian siang dan malam, matahari, dan terciptanya langit dan bumi adalah tantangan bagi orang beriman dan intelektual untuk mengakui kebesaran Tuhan. Ayat dalam surat Al-'Imran ayat 190-191 menyatakan bahwa orang beriman, setelah merenungkan

keindahan alam semesta, akan secara langsung melakukan dzikir dan meyakini bahwa setiap ciptaan Allah SWT memiliki berbagai manfaat. Dengan demikian, saat melakukan perjalanan wisata, seseorang akan merenungkan keajaiban ciptaan Allah SWT, bersyukur atas ciptaan-Nya, dan memanfaatkannya.

Pariwisata kini telah menjadi bagian integral dalam kehidupan manusia. Sektor pariwisata telah membuat beberapa wilayah di Indonesia mengadopsi pariwisata sebagai keunikan daerah mereka. Salah satu contohnya adalah Malang Raya, yang terdiri dari Kabupaten Malang, Kota Malang, dan Kota Batu (Anton, 2015). Malang Raya dikenal sebagai destinasi wisata utama di Provinsi Jawa Timur dan Indonesia. Berdasarkan data statistik Kementerian Pariwisata dan Ekonomi Kreatif tahun 2022, jumlah wisatawan mancanegara yang mengunjungi Malang Raya mencapai 6 juta wisatawan (Kemenparekraf/Baparekraf, 2022).

Kawasan Malang Raya menawarkan berbagai macam keindahan destinasi wisata yang memiliki ciri khas yang berbeda dengan daerah lain di Indonesia. Wisata alam dan buatan manusia menjadi beberapa destinasi wisata yang di tawarkan oleh Malang Raya (Maryono, 2016). Menurut data dari Kabupaten Malang Satu Data, di wilayah Kabupaten Malang terdapat 16 objek desa wisata, 106 objek wisata alam, 49 objek wisata budaya, dan 24 objek wisata buatan (Pemerintah Kabupaten Malang, 2018). Menurut data dari Badan Pusat Statistik (BPS) Kota Batu, di wilayah Kota Batu terdapat 10 objek wisata buatan, 12 objek desa wisata, 5 objek wisata alam, 5 objek wisata oleh-oleh, dan 1 objek wisata religi (Badan Pusat Statistik Kota Batu, 2021). Menurut data dari Malang Satu Data, di wilayah Kota Malang terdapat 16 objek wisata budaya, 2 objek wisata sejarah, 4

objek wisata religi, 1 objek wisata pendidikan, 2015 objek wisata kuliner, 12 objek wisata belanja, dan 20 objek wisata buatan (Pemerintah Kota Malang, 2021).

Banyaknya tujuan destinasi wisata di kawasan Malang Raya menjadi daya tarik tersendiri bagi para wisatawan mancanegara. Namun, seiring dengan bertambahnya destinasi wisata, muncul permasalahan terkait informasi mengenai destinasi tersebut. Wisatawan mancanegara membutuhkan informasi yang komprehensif mengenai rute perjalanan mereka, namun tidak semua sumber informasi seperti media cetak, televisi, dan internet dapat memenuhi kebutuhan ini (Putra, 2017).

Tantangan lain di sektor pariwisata adalah rendahnya kualitas layanan dan kurangnya jumlah sumber daya manusia di industri jasa pariwisata. Hal ini perlu mendapatkan perhatian khusus untuk meningkatkan sektor pariwisata di kawasan Malang Raya. Kualitas layanan pariwisata dan jumlah sumber daya manusia menjadi standar perbandingan untuk mencapai kepuasan wisatawan (Wirajaya, 2013). Peningkatan dalam kualitas layanan dan jumlah sumber daya manusia akan berdampak pada kualitas layanan sosial dan informasi yang diberikan kepada wisatawan mancanegara, sehingga dapat meningkatkan kualitas layanan informasi mengenai destinasi wisata di Kawasan Malang Raya.

Persoalan lain ditemui pada kondisi *website* resmi Dinas Kebudayaan dan Pariwisata Kabupaten Malang, Kota Malang, dan Kota Batu saat ini yang hanya menyediakan informasi singkat tentang objek wisata saja dan tidak ada fitur tanya jawab secara interaktif yang bisa memberitahu para wisatawan asing akses menuju ke destinasi wisata (Maryono, 2016). Akibatnya, wisatawan asing harus melakukan

pencarian rute atau akses sendiri melalui beberapa tahap. Kurangnya layanan informasi digital seperti ini bisa mengakibatkan kurangnya efisiensi dan efektifitas dalam mendapatkan informasi akses atau rute menuju destinasi tujuan yang telah dipilih oleh wisatawan mancanegara (Wirajaya, 2013).

Persoalan lain ditemui adalah komunikasi antar warga lokal dan wisatawan mancanegara sangat kurang sekali. Hal ini dikarenakan karena minimnya pengetahuan yang menyebabkan tidak semua warga lokal mampu memahami dan mengerti tentang apa yang wisatawan mancanegara komunikasikan (Nugroho, 2020). Terlebih lagi apabila para wisatawan mancanegara berkomunikasi menggunakan bahasa global yaitu menggunakan Bahasa Inggris. Hal ini yang perlu menjadi perhatian utama bagi pemerintah khususnya bagi Lembaga Sektor Pariwisata di Kawasan Malang Raya.

Melihat *urgensi* sulitnya wisatawan mancanegara dalam mendapatkan informasi terkait rute menuju ke destinasi wisata yang dituju khususnya di Jawa Timur dikarenakan faktor informasi yang terbatas dari badan pemerintah yang menangani sektor pariwisata hingga faktor sulitnya komunikasi kepada warga lokal yang tidak semuanya bisa mengerti apa yang dikomunikasikan oleh para wisatawan mancanegara. Menurut penulis, perlunya *chatbot* interaktif yang dapat membantu dalam mendapatkan informasi rute dan jalan akses menuju destinasi wisata tujuan yang diinginkan akan memudahkan wisatawan mancanegara.

Dalam perspektif Islam, penyampaian informasi tidak hanya dipandang sebagai aktivitas duniawi, melainkan juga memiliki nilai ibadah ketika dilakukan dengan cara yang benar. Dalam muamalah, informasi adalah instrumen amal yang

bermanfaat jika dijalankan sesuai etika syariah. Memberikan informasi termasuk praktik sosial yang mencerminkan hubungan dengan Allah SWT dan sesama manusia. Disebut hubungan dengan Allah SWT (Ma'Allah), ketika informasi disampaikan dengan niat tulus dan ikhlas semata karena Allah SWT, artinya setiap bentuk komunikasi yang berlandaskan kejujuran dan ketulusan dapat menjadi bagian dari amal saleh yang bernilai ibadah. Di sisi lain, hubungan dengan sesama manusia (Ma'annas) menuntut agar informasi yang diberikan benar, jelas, dan tidak merugikan orang lain. Dengan demikian, penyampaian informasi dalam Islam tidak hanya memperhatikan aspek teknis, tetapi juga aspek spiritual dan moral.

Adapun islam sangat menghargai tata krama yang baik dalam menyampaikan informasi. Al-Qur'an memberikan peringatan bahkan memerintahkan untuk melakukan tradisi *tabayyun* (pemeriksaan ulang) ketika ada informasi, dengan tujuan agar tidak menimbulkan kerugian pada orang lain. Allah berfirman di dalam Alqur'an surat Al-Hujuraat ayat 6 yang berbunyi:

يَٰٓأَيُّهَا ٱلَّذِينَ ءَامَنُوٓاْ إِن جَآءَكُمْ فَاسِقٌۢ بِنَبَإٖ فَتَبَيَّنُوٓاْ أَن تُصِيبُواْ قَوْمَۢا بِجَهَٰلَةٖ فَتُصْبِحُواْ عَلَىٰ مَا فَعَلْتُمْ نَٰدِمِينَ

"*Hai orang-orang yang beriman, jika datang kepadamu orang fasik membawa suatu berita, Maka periksalah dengan teliti agar kamu tidak menimpakan suatu musibah kepada suatu kaum tanpa mengetahui keadaannya yang menyebabkan kamu menyesal atas perbuatanmu itu. (6)*" Q.S. Al-Hujurat : 6)

Menurut tafsir Ibnu Katsir menjelaskan bahwa Allah memperingatkan kaum Mukminin agar tidak terburu-buru menerima berita dari fasik tanpa melakukan penelitian. Orang yang fasik tidak dapat dipercaya, dan menerima berita tanpa verifikasi dapat menimbulkan kerugian jiwa dan harta. Pedoman ini ditujukan untuk

menghindari penyesalan di kemudian hari, dengan menekankan perlunya penelitian sebelum mempercayai suatu berita, terutama jika sumbernya tidak dapat dipercaya.

Ayat tersebut menguraikan pentingnya berhati-hati dalam memberikan suatu informasi. Pemberian informasi yang tidak akurat dapat menimbulkan kerugian bagi penerima informasi tersebut. Oleh karena itu, fungsi *chatbot* sebagai penyedia informasi dengan menggunakan data yang valid dapat memberikan informasi yang tepat dan akurat kepada pengguna yang memerlukan.

Penelitian terdahulu yang dilakukan oleh Maskur dengan mengambil judul "Perancangan Chatbot Pusat Informasi Mahasiswa Menggunakan AIML Sebagai *Virtual Assistant* Berbasis Web". Dari penelitian ini, mendapatkan hasil pemanfaatan *chatbot* yang telah dilengkapi dengan kecerdasan buatan secara cepat dengan ketepatan jawaban hanya mencapai 60% (Maskur, 2016). Penelitian terdahulu yang dilakukan oleh Zifora Nur Baiti dengan mengambil judul "Aplikasi *Chatbot* MI3 Untuk Informasi Jurusan Teknik Informatika Berbasis Sistem Pakar Menggunakan Metode *Forward Chaining*". Dari penelitian ini, mendapatkan hasil keberhasilan dalam pemanfaatan aplikasi *chatbot* sebesar 92,5% dengan tingkat keakurasian mencapai 76% (Baiti, 2013).

Untuk meningkatkan akurasi dalam memprediksi pemilihan kalimat, diperlukan penerapan suatu metode. Metode yang digunakan dalam penelitian ini adalah *Random Forest Classifier*, sebuah algoritma yang dihasilkan dari proses *boostrap aggregating* pada algoritma *Decision Tree.* Pemilihan metode ini didasarkan pada keunggulannya dibandingkan dengan metode algoritma lainnya, terutama karena masuk ke dalam kelompok metode *Classification and Regression*

Tree (CART), yakni metode klasifikasi yang memanfaatkan data historis untuk pembentukan struktur keputusan. Dari hasil penelitian ini diharapkan *chatbot* "Gumara" dengan menggunakan metode *Random Forest Classifier* dapat mengoptimalkan tingkat keakurasian dalam kinerja prediksi kalimat dan menggunakan telegram *messenger* untuk membuat data lebih terstruktur dan juga memberikan *social services*.

1.2 Pernyataan Masalah

Berdasarkan uraian diatas, maka rumusan masalah dari penelitian ini diantaranya adalah:

1. Bagaimana perancangan dan pembangunan *Chatbot Virtual Route Guide* "Gumara" menggunakan Metode *Random Forest Classifier?*
2. Bagaimana hasil akurasi pada *Chatbot Virtual Route Guide* "Gumara" menggunakan metode *Random Forest Classifier*?

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai melalui penelitian ini diantaranya adalah:

1. Merancang dan membangun *Chatbot Virtual Route Guide* berbasis *telegram messenger* dengan menggunakan Metode *Random Forest Classifier* untuk mengetahui informasi mengenai rute menuju destinasi wisata bagi wisatawan mancanegara.
2. Menganalisa hasil akurasi *Chatbot Virtual Route Guide* "Gumara" menggunakan metode *Random Forest Classifier.*

1.4 Batasan Masalah

Untuk menjaga agar pembahasan penelitian ini tetap sesuai dengan rumusan yang telah ditetapkan, maka perlu adanya pembatasan-pembatasan, di antaranya.:

1. Bahasa yang digunakan dalam percakapan *chatbot virtual route guide* hanya menggunakan Bahasa Inggris.
2. *Chatbot virtual route guide* tidak melayani input dalam bentuk perhitungan matematis, dan tidak menanggapi input yang berupa karakter.
3. Fokus pada pengembangan dan implementasi *chatbot virtual route guide* hanya untuk platform telegram menggunakan fitur dan *API* yang tersedia di telegram.

1.5 Manfaat Penelitian

Dalam penerapan metode *Random Forest Classifier* pada *chatbot* "Gumara" sebagai *virtual tour guide* diharapkan mampu memberikan kegunaan yang bermanfaat, antara lain:

1. Bagi peneliti
    a. Dapat meningkatkan pengetahuan terkait kecerdasan buatan khususnya pada penerapan *chatbot*.
    b. Dapat menerapkan pengetahuan yang di dapat di bangku perkuliahan agar dapat diimplementasikansss dengan baik.
2. Bagi peningkatan nilai ekonomi

    Menambah pendapatan ekonomi masyarakat sekitar dengan banyaknya wisatawan yang berkunjung setelah menggunakan *chatbot* "Gumara".
3. Bagi jurusan Teknik Informatika

Hasil penelitian diharapkan dapat digunakan sebagai sarana pengenalan dan referensi mengenai perancangan *chatbot* pada perpustakaan Universitas Islam Negeri Maulana Malik Ibrahim Malang khususnya di Jurusan Teknik Informatika.

BAB II

TINJAUAN PUSTAKA

2.1 Studi Literatur

Penelitian dengan topik *chatbot* dilakukan oleh Abilowo dkk. (2020) membahas perancangan *chatbot* berbasis *Artificial Intelligence Markup Language* (AIML) untuk mempelajari bahasa Jawa. Tujuan dari penelitian ini untuk mendukung pelestarian bahasa tersebut melalui teknologi. *Chatbot* ini dibuat untuk menyajikan materi pada tiga tingkatan bahasa Jawa dan mengajukan pertanyaan kepada pengguna. Meskipun chatbot berfungsi baik dengan tingkat akurasi 90%, penelitian mengidentifikasi kekurangan terkait kompleksitas algoritma dan potensi *overfitting* pada penerapan *Natural Language Processing* jika data terlalu banyak (Abilowo et. al,. 2020).

Penelitian oleh Maskur (2016) berjudul "Perancangan *Chatbot* Pusat Informasi Mahasiswa Menggunakan AIML Sebagai *Virtual Assistant* Berbasis Web" mengeksplorasi penerapan *Artificial Intelligence Markup Language* (AIML) dalam *chatbot* sebagai asisten virtual untuk memberikan informasi kepada mahasiswa berdasarkan data sistem, terutama terkait program studi teknik informatika. Hasil penelitian menunjukkan bahwa penggunaan AIML dalam chatbot memungkinkan integrasi input teks menghasilkan percakapan antara pengguna dan program dengan tingkat ketepatan jawaban mencapai 60% (Maskur, 2016).

Dalam penelitian Zifora Nur Baiti (2013) tentang aplikasi *chatbot* "MI3" untuk informasi jurusan Teknik Informatika di UIN Maliki Malang berbasis sistem pakar dengan metode *forward chaining*, ditemukan bahwa *chatbot* ini mampu mengenali kata kunci dalam kalimat-kalimat terkait dengan kategori tertentu. Meskipun aplikasi ini berhasil mendapatkan persentase Sangat Setuju sebesar 48,88% dan Setuju sebesar 51,22% dari 25 responden. Pada penelitian ini terdapat kekurangan yang perlu diatasi, yaitu melengkapi data sebagai kata kunci agar sistem lebih mudah mengenali kata pada pertanyaan. Kelemahan ini disebabkan oleh kebutuhan metode *forward chaining* yang memerlukan data kata kunci yang cukup banyak untuk menjawab pertanyaan dengan benar (Baiti, 2013).

Penelitian Damar Adyun Muhamad (2020) membahas pembangunan sistem deteksi untuk membedakan serangan DDoS dan akses normal menggunakan metode *Random Forest*. Hasil pengujian menunjukkan tingkat akurasi, presisi, *recall*, dan *f-measure* yang tinggi pada dua jenis set data, yaitu set data primer dari simulasi serangan DDoS peneliti dan set data sekunder CICIDS2017 dari University of New Brunswick Kanada, dengan durasi pengujian yang singkat. Pada set data primer, metode *Random Forest* mencapai nilai akurasi 99.62%, presisi 98.59%, *recall* 100%, dan *f-measure* 99.29% dalam waktu 1 detik. Sementara pada set data sekunder, nilai akurasi, presisi, *recall*, dan *f-measure* berturut-turut adalah 99.87%, 99.91%, 99.86%, dan 99.89%, dengan durasi pengujian selama 12 menit 44 detik (Muhamad, 2020).

Penelitian yang dilakukan oleh Akhmad Syukron dan Agus Subekti (2018) membahas penerapan metode *Random Over-Under Sampling* dan *Random Forest*

dalam klasifikasi penilaian kredit terhadap calon debitur. Permasalahan utama yang diatasi adalah ketidakseimbangan distribusi dataset pada dataset German Credit. Hasil pengujian menunjukkan bahwa metode *Random Forest* menghasilkan nilai akurasi 76%, sedangkan penerapan metode *Random Over-Under Sampling Random Forest* meningkatkan akurasi hingga 90,1%, membuktikan peningkatan signifikan dalam klasifikasi penilaian kredit (Syukron & Subekti, 2018).

Siska Devella, Yohannes, dan Firda Novia Rahmawati (2020) dalam penelitiannya "Implementasi *Random Forest* Untuk Klasifikasi Motif Songket Palembang Berdasarkan *SIFT*" mengklasifikasikan citra motif Songket Palembang dengan menggunakan ekstraksi fitur *Scale-Invariant Feature Transform (SIFT)*. Proses pembentukan fitur menggunakan metode *SIFT* melalui tahapan *extrema detection scale space*, *keypoint localization*, *orientation assignment*, dan *keypoint descriptor*. Fitur yang dihasilkan digunakan untuk klasifikasi *Random Forest*. Gambar motif songket yang digunakan dalam penelitian ini sebanyak 115 gambar dari masing-masing jenis motif, yaitu Bunga Cina, Bunga Indah, dan Pulir. Pemilihan gambar diambil dari 5 warna masing-masing motif Songket Palembang. Data latih dan data uji yang digunakan masing-masing adalah 100 dan 15 untuk masing-masing motif Songket Palembang. Hasil pengujian menunjukkan bahwa metode *SIFT* dan *Random Forest* untuk klasifikasi motif Songket Palembang dapat memberikan akurasi yang cukup baik sebesar 92,98%, akurasi per kelas sebesar 94,07%, presisi sebesar 92,98 %, dan ingat 89,74%.

Tabel 2. 1 Studi Literatur

No	Judul Penelitian	Nama dan Tahun	Metode Penelitian	Hasil Penelitian
1	Perancangan *Chatbot* Sebagai Pembelajaran Dasar Bahasa Jawa Menggunakan *Artificial Intelligence Markup Language*	Krisanto Abilowo, Mayanda Mega Santoni, dan Anita Muliawati (2020)	*Artificial Intelligence Markup Language* (AIML)	Hasil dari penelitian ini adalah sistem *chatbot* mampu berfungsi dengan baik karena memiliki akurasi sebesar 90%.
2	Perancangan *Chatbot* Pusat Informasi Mahasiswa Menggunakan AIML Sebagai *Virtual Assistant* Berbasis Web	Maskur (2016)	*Artifical Intelligence Markup Language* (AIML)	Hasil pemanfaatan *chatbot* yang telah dilengkapi dengan kecerdasan buatan secara cepat dengan ketepatan jawaban hanya mencapai 60%.
3	Aplikasi *Chatbot* MI3 Untuk Informasi Jurusan Teknik Informatika Berbasis Sistem Pakar Menggunakan Metode *Forward Chaining*	Zifora Nur Baiti (2013)	*Forward Chaining*	Hasil penelitian dengan menginputkan kalimat-kalimat yang berhubungan dan tidak dengan kategori, aplikasi ini mampu mengenali kata kunci pada kalimat-kalimat tersebut. Hal ini mengacu pada hasil pengujian yang didapatkan persentase sebesar Sangat Setuju 48,88% dan Setuju 51,22% dari 25 responden dengan beberapa kalimat inputan.

4	Deteksi *Distributed Denial Of Service* Menggunakan *Random Forest*	Damar Adyun Muhamad (2020)	*Random Forest*	Hasil pengujian pada 266 baris set data primer, didapatkan nilai *accuracy*, *presicion*, *recall* dan *f-measure* secara berturut-turut adalah 99.62%, 98.59%, 100% dan 99.29% dengan durasi pengujian selama 1 detik. sedangkan hasil pengujian pada 169309 baris set data sekunder didapatkan nilai *accuracy*, *presicion*, *recall* dan *f-measure* secara berturut-turut adalah 99.87%, 99.91%, 99.86% dan 99.89% dengan durasi pengujian selama 12 menit 44 detik.
5	Penerapan Metode *Random Over-Under Sampling* dan *Random Forest* Untuk Klasifikasi Penilaian Kredit	Akhmad Syukron dan Agus Subekti (2018)	*Random Over-Under Sampling* dan *Random Forest*	Hasil pengujian menunjukan bahwa klasifikasi tanpa melalui proses *resampling* menghasilkan kinerja akurasi rata-rata 70 % pada semua *classifier*. Metode Random Forest memiliki nilai akurasi yang lebih baik dibandingkan dengan beberapa metode lainnya dengan nilai akurasi sebesar 0,76 atau 76%. Sedangkan klasifikasi dengan penerapan metode *Random Over-under sampling Random Forest* dapat meningkatkan kinerja akurasi

				sebesar 14,1% dengan nilai akurasi sebesar 0,901 atau 90,1 %.
6	Implementasi *Random Forest* Untuk Klasifikasi Motif Songket Palembang Berdasarkan *SIFT*	Siska Devella, Yohannes, dan Firda Novia Rahmawati (2020)	*Random Forest*	Hasil pengujian menunjukkan bahwa metode *SIFT* dan *Random Forest* untuk klasifikasi motif Songket Palembang dapat memberikan akurasi yang cukup baik, dimana metode *SIFT* dan *Random Forest* menghasilkan akurasi keseluruhan sebesar 92,98%, *accuracy* per kelas sebesar 94,07%, *precision* sebesar 92,98 %, dan *recall* 89,74%.

Hasil studi literatur pada tabel 2.1 menunjukkan bahwa metode *Random Forest* dapat meningkatkan hasil akurasi dibandingkan dengan beberapa metode seperti metode *Forward Chaining* dan *Artificial Intelligence Markup Language* (AIML). Penggunaan metode kecerdasan buatan *Random Forest Classifier* dapat meningkatkan hasil keakurasian dalam jumlah dataset yang besar. Algoritma *Random Forest Classifier* adalah pengembangan dari model algoritma *Decision Tree* dimana setiap pohon fikiran dilatih dengan sampel individu. Metode *Random Forest* dapat meningkatkan hasil akurasi karena dalam membangkitkan simpul anak untuk setiap *node* dilakukan secara acak. Metode ini digunakan untuk membangun *decision tree* yang terdiri dari *root node*, *internal node*, dan *leaf node* dengan mengambil atribut data secara acak sesuai ketentuan yang diberlakukan.

2.2 Landasan Teori

Landasan teori merupakan pernyataan yang berisikan tentang teori-teori yang memberikan pemahaman dan konsep yang berhubungan dengan topik penelitian.

2.2.1 Pariwisata

Pariwisata adalah perjalanan dari satu tempat ketempat lain bersifat sementara, dilakukan perorangan atau kelompok, sebagai usaha mencari keseimbangan atau keserasian dan kebahagiaan dengan lingkungan dalam dimensi sosial budaya, alam, dan ilmu (Yuda Wenika, 2014). Pariwisata, yang berasal dari akar kata wisata menurut UU Republik Indonesia No. 9 Tahun 1990 tentang kepariwisataan mendefinisikan wisata sebagai kegiatan perjalanan yang dilakukan oleh seseorang atau sekelompok orang mengunjungi tempat tertentu dengan tujuan rekreasi, mengembangkan pribadi, atau mempelajari daya tarik wisata yang dikunjungi.

2.2.2 *Chatbot*

Chatbot merupakan program dalam kecerdasan buatan yang diciptakan untuk berinteraksi langsung dengan manusia sebagai pengguna. Perbedaan mendasar antara *chatbot* dan sistem pemrosesan bahasa alami (*Natural Language Processing System*) terletak pada kesederhanaan algoritma yang digunakan. Meskipun banyak *bot* mampu menginterpretasikan dan menanggapi input manusia, sebenarnya *bot* tersebut hanya memahami kata kunci dalam input dan memberikan respon berdasarkan kata kunci yang paling relevan atau pola kata yang paling mirip

dari data yang telah tersimpan dalam database yang telah dibuat sebelumnya (Weizenbaum, 1966).

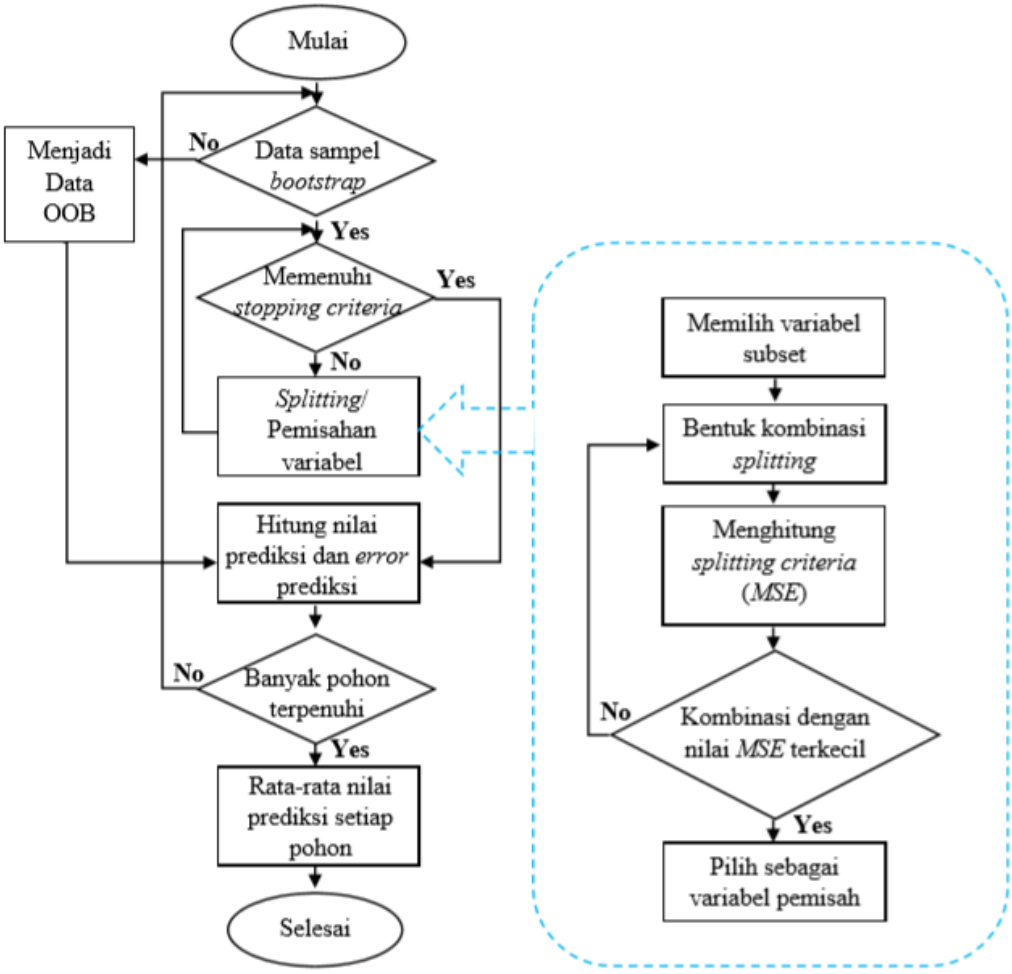
Pada mulanya program komputer (*bot*) ini diuji melalui *turing test,* yaitu dengan merahasiakan identitasnya sebagai mesin sehingga dapat membohongi orang yang berbicara dengannya. Jika pengguna tidak dapat mengidentifikasi *bot* sebagai suatu program komputer, maka *chatbot* tersebut dikategorikan sebagai kecerdasan buatan (Dana Vrajitoru, 2004).

Salah satu *chatbot* yang terkenal adalah Eliza (Dr. Eliza) yang dikembangkan oleh Joseph Weizenbaum di Massachusetts Institute of Technology (MIT). Eliza adalah pelopor *chatbot* yang dikenal sebagai program *chat* yang memiliki profesi sebagai seorang psikiater. Eliza mensimulasikan percakapan antara seorang psikiater dengan pasiennya dalam bahasa Inggris yang alami. Eliza dibuat dengan tujuan untuk mempelajari komunikasi *natural language* antara manusia dengan mesin. Eliza bertindak seolah-olah dia adalah seorang psikolog yang dapat menjawab pertanyaan pertanyaan dari pasien dengan jawaban yang cukup masuk akal atau menjawabnya dengan pertanyaan balik (Weizenbaum, 1966).

2.2.3 *Random Forest Classifier*

Random Forest merupakan metode yang diperkenalkan oleh Breiman. Sebagai pengembangan dan kombinasi dari banyak *Decision Tree*. Jika pada *Decision Tree* merupakan pohon klasifikasi tunggal maka pada *Random Forest* dibuat banyak pohon untuk menentukan hasil prediksinya. Kombinasi dari *bootstrap aggegating* dan *random feature selection* pada *Random Forest* dapat

digunakan untuk mengurangi masalah *overfitting* pada data latih yang kecil (Breiman et al., 2017). Dikarenakan *Random Forest* merupakan metode *ensemble* dari CART maka pada *Random Forest* juga tidak memiliki asumsi atau baik digunakan pada kasus non parametrik.



Gambar 2. 1 Flowchart *Random Forest*

Langkah-langkah yang dilakukan pada *random forest* yaitu:

1. Menentukan parameter *Random Forest*. Nilai *mtry* atau jumlah variabel prediktor yang diambil secara acak sebanyak $\frac{p}{3}$ untuk kasus regresi dimana p adalah jumlah seluruh variabel prediktor (Breiman et al., 2017).

2. Kemudian menentukan N_{tree} pohon yang disarankan untuk digunakan yaitu sebanyak 50 pohon. Berdasarkan Breiman (2017) mengatakan bahwa 50 pohon sudah memberikan hasil yang memuaskan untuk kasus klasifikasi. Sedangkan Sutton (2005), menyatakan bahwa $N_{tree} \geq 100$ menghasilkan misklasifikasi yang cenderung rendah.
3. Menentukan *stopping criteria default* pada *scikit-learn Random Forest* yaitu 1, artinya jika didalam *sub-node* / simpul anak hanya terdapat 1 sampel maka *subnode* tersebut akan berhenti melakukan *splitting* sehingga menjadikan *sub-node* tersebut menjadi terminal *node*/*leaf node*. Maka jika percabangan sudah berhenti akan dihasilkan simpul terminal sebagai hasil prediksi satu pohon CART (Widmaier, 2015).
4. Membagi data menjadi data latih dan data uji. Dari data latih diambil *n* sampel dengan pengembalian (*bootstrap*) sehingga diperoleh dataset baru Di dimana *i* adalah pembagian sampel *bootstrap* pohon ke-*i*. Pengambilan sampel *bootsrap* hanyalah 2/3 dari seluruh data latih, 1/3 data yang tidak terambil pada *bootstrap* maka akan dijadikan data *out-of-bag*, data *out-of-bag* berguna untuk mengukur performa pohon regresi (Liu et al., 2013).
5. Melakukan prediksi berdasarkan pembentukan model pohon dari *dataset* baru Di dengan kombinasi dari *m* variabel prediktor yang diambil secara acak (*random feature selection*).
6. Tahap pembentukan pohon pada *Random Forest* pada hasil *bootstrap* yaitu:
   6.1. Setelah menentukan sampel berdasarkan hasil dari *bootstrap* maka langkah selanjutnya yaitu menentukan variabel *root node*/simpul akar

yaitu variabel penjelas yang berada paling atas untuk dijadikan variabel pemisah/*splitting attribute*. Pemilihan *splitting attribute* ini diambil berdasarkan *splitiing criteria*/kriteria pemisahan. Ada banyak kriteria pemisahan yang umum digunakan yaitu nilai *entropy*/*information gain*, nilai *gini*, ataupun MSE (*Mean Square Error*). *Splitting criteria* yang digunakan pada *Random Forest* yaitu menggunakan MSE yang diterapkan pada *package python scikit-learn*. Variabel penjelas yang memiliki nilai MSE terkecil maka akan mendapatkan kesempatan lebih besar untuk menjadi variabel pemisah (Widmaier, 2015). Secara matematis MSE dinyatakan pada Persamaan 2.1.

$$MSE_n = \frac{1}{N} + \sum_{i=1}^{N} (Y_i - \bar{Y}_n)^2 \qquad (2.1)$$

Dimana:

MSE_n : Nilai *MSE* pada pohon ke-*n*

N : Jumlah sampel pada pohon ke-*n*

Y_i : Nilai sampel ke-*i* pada pohon ke-*n*

$\bar{Y}_n$: Nilai rata-rata sampel pohon ke-*n*

6.2. Setelah mendapatkan atribut pemisah dengan nilai *MSE* terendah maka akan dibuat cabang selanjutnya dengan menentukan variabel pemisah berikutnya yang dinamakan *sub-node*/simpul anak.

6.3. Hal tersebut terus dilakukan sampai mencapai *stopping criteria* yaitu minimum sampel per simpul *terminal*/*terminal node*/*leaf node* terpenuhi.

7. Mengulangi langkah 1 sampai 3 hingga diperoleh sebanyak N_{tree} pohon yang diinginkan.
8. Menentukan hasil prediksi akhir dengan cara hasil prediksi pada setiap pohon akan digabungkan (*aggregate*). Nilai prediksi pada kasus klasifikasi dipertimbangkan berdasarkan *majority vote* (*vote* terbanyak pada pohon klasifikasi) dan nilai prediksi pada kasus regresi diambil dari nilai rata-rata dari setiap pohon (Criminisi et al., 2012). Secara matemastis nilai rata-rata seluruh prediksi pohon *CART* pada *random forest* pada Persamaan 2.2.

$$\hat{Y}_i = \frac{1}{N_{tree}} + \sum_{n=1}^{N_{tree}} \hat{Y}_n \tag{2.2}$$

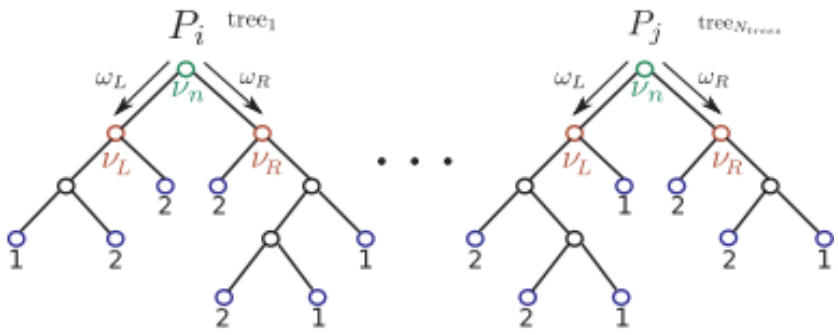
Dimana:

$\hat{Y}_i$: Hasil prediksi akhir

N_{tree} : Total jumlah pohon pada *random forest*

$\hat{Y}_n$: Hasil prediksi pokon ke-*n*

9. Menentukan nilai akurasi model *random forest* menggunakan nilai akurasi.



Gambar 2. 2 Pohon *Random Forest*

Pada Gambar 2.2 merupakan representasi dari pohon *Random Forest* yang merupakan kumpulan dari pohon-pohon *CART*. P_i merupakan data sampel

bootstrap pada pohon ke-*i* yang diambil dari *data training* (*P*). Simpul akar/ *root node* (V_n) dilakukan *splitting* menjadi dua cabang berdasarkan *splitting criteria*, data *subset* yang memenuhi kriteria akan masuk ke dalam cabang *left branch* (ω_L) yang kemudian akan diperoleh simpul anak/ *child node*/ *sub node* ((V_L)) dan akan terus membagi hingga mecapai simpul *terminal*/ *terminal node*/*leaf node* (simpul akhir berwarna biru) (Izquierdo-Verdiguier & Zurita-Milla, 2020). Hal tersebut dilakukan terus menerus hingga sebanyak N_{tree} pohon yang sudah ditentukan.

2.2.4 *Telegram Messenger*

Menurut (Pinto, 2014) Telegram sebagai salah satu aplikasi pesan instan, mengklaim dapat menutupi beberapa kekurangan yang ada pada Whatsapp. Telegram merupakan aplikasi *cloud based* dan alat enkripsi. Telegram menyediakan enkripsi *end-to-end*, *self destruction messages*, dan infrastruktur *multi-data center*. Di United States dan beberapa Negara lainya, Telegram menjadi aplikasi nomor satu untuk kategori *social networking*, didepan Facebook, Whatsapp, Kik, dll (Hamburger & Elise, 2014).

Dalam website resminya, Telegram dikembangkan oleh Pavel dan Nikolai Durov, sebagian besar pengembangan Telegram awalnya berada di St. Petersburg namun tim Telegram harus meninggalkan Rusia karena peraturan mengenai kebijakan tentang Teknologi Informasi setempat. Maka untuk saat ini Telegram berbasis di Dubai. Akun resmi twitter Telegram @telegram menyatakan bahwa awal 2018 memiliki lebih dari 100 juta pengguna aktif. Adapun keunggulan fitur Telegram Messenger sebagai berikut :

1. Privasi, pesan telegram sangat terenkripsi dan dapat membuat pengguna terjamin keamanan serta privasinya.
2. Cepat, telegram mengirimkan pesan lebih cepat daripada aplikasi lainnya.
3. Terdistribusi, server telegram tersebar di seluruh dunia untuk keamanan dan kecepatan.
4. Gratis, yang dimaksud gratis dalam pengertiannya yaitu telegram bebas iklan dan akan gratis selamanya serta tanpa biaya berlangganan.
5. Aman, telegram menjamin akan selalu menjaga keamanan pesan dari serangan *hacker*.
6. Powerful, telegram tidak memiliki batas pada ukuran media serta pesan yang kita kirim.

Telegram memberikan kemudahan akses bagi pengguna karena tersedia pada platform mobile maupun desktop. Pada platform mobile Telegram dapat digunakan di iPhone/iPad, Android dan Windows phone, sedangkan pada platform desktop Telegram dapat digunakan di Windows, Linux, Mac OS dan juga Web-browser. (Hamburger & Elise, 2014) juga menambahkan Telegram mengklaim sebagai aplikasi pesan massal tercepat dan teraman yang berada di pasar. Telegram pertama kali diluncurkan hanya untuk platfrom iOS pada tanggal 14 Agustus 2013. Sedangkan versi alfa dari Telegram untuk platfrom Android diluncurkan secara resmi pada tanggal 20 Oktober 2013.

2.2.5 Telegram *Bot*

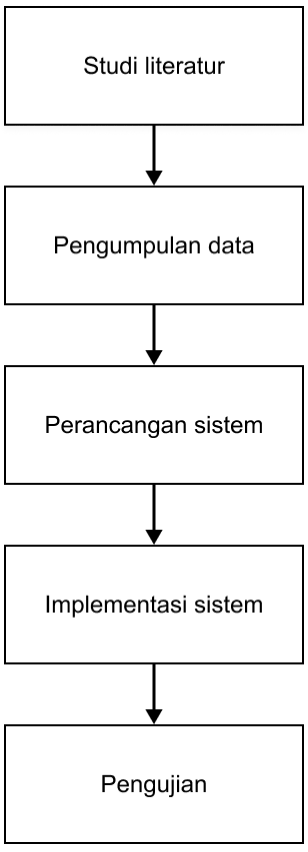
Telegram *bot* merupakan akun Telegram khusus yang didesain dapat menangani pesan secara otomatis. Pengguna dapat berinteraksi dengan *bot* dengan

mengirimkan pesan perintah (*command*) melalui pesan privasi maupun grub. Akun Telegram *bot* tidak memerlukan tambahan nomor telepon pada penbuatannya. Akun ini hanya bertugas sebagai antarmuka dari kode yang berjalan di sebuah server. Telegram *bot* dapat dibangun sesuai dengan kebutuhan, semisal digunakan dengan mengintegrasikannya ke layanan lain untuk mengendalikan *smart home*, membangun *social services*, membangun *custom tools*, ataupun melakukan hal lain secara *virtual*.

BAB III

METODOLOGI PENELITIAN

Pada bab ini akan dibahas mengenai prosedur atau tahapan penelitian yang akan dilakukan. Prosedur penelitian adalah langkah pada penelitian ini yang dilaksanakan berdasarkan pendekatan baku yang mencakup cara-cara dan teknik dalam pengumpulkan data dan menjawab pertanyaan dalam penelitian ini. Tahapan penelitian terdiri dari studi literatur, pengumpulan data, perancangan sistem, implementasi sistem, hasil uji coba dan penarikan kesimpulan. Prosedur dalam penelitian ini direpresentasikan pada Gambar 3.1.



Gambar 3. 1 Tahapan Penelitian

3.1 Studi Literatur

Tahapan studi literatur dilakukan agar memperoleh referensi dan referensi sebanyak-banyaknya yang berkaitaan dengan lingkup pembahasan pada penelitian. Dalam tahap ini dilakukan mempelajari literatur yang berkaitan dengan Rancang Bangun *Chatbot* "Gumara" Sebagai *Virtual Route Guide* Wisatawan Asing Menggunakan Metode *Random Forest Classifier*, diantaranya:

a. Pariwisata
b. *Chatbot*
c. Random Forest
d. *Telegram messenger*
e. Algoritma *Random Forest Classifier*

3.2 Pengumpulan Data

Pengumpulan data dilakukan untuk memperoleh data yang dibutuhkan dalam rangka mencapai tujuan penelitian. Penulis menggunakan data primer berupa dataset kalimat percakapan dalam bahasa Inggris yang didapatkan dari platform Hugging Face. Dataset tersebut berisi kalimat percakapan berupa kalimat sapaan, kalimat pernyataan, dan kalimat pertanyaan. Dataset komentar yang baru didapatkan masih berstatus data sekunder karena penulis belum mengolah data tersebut. Data sekunder akan diubah menjadi data primer dengan dilabeli manual oleh tenaga ahli. Data primer digunakan sebagai masukan untuk membangun model klasifikasi dan menguji proses klasifikasi.

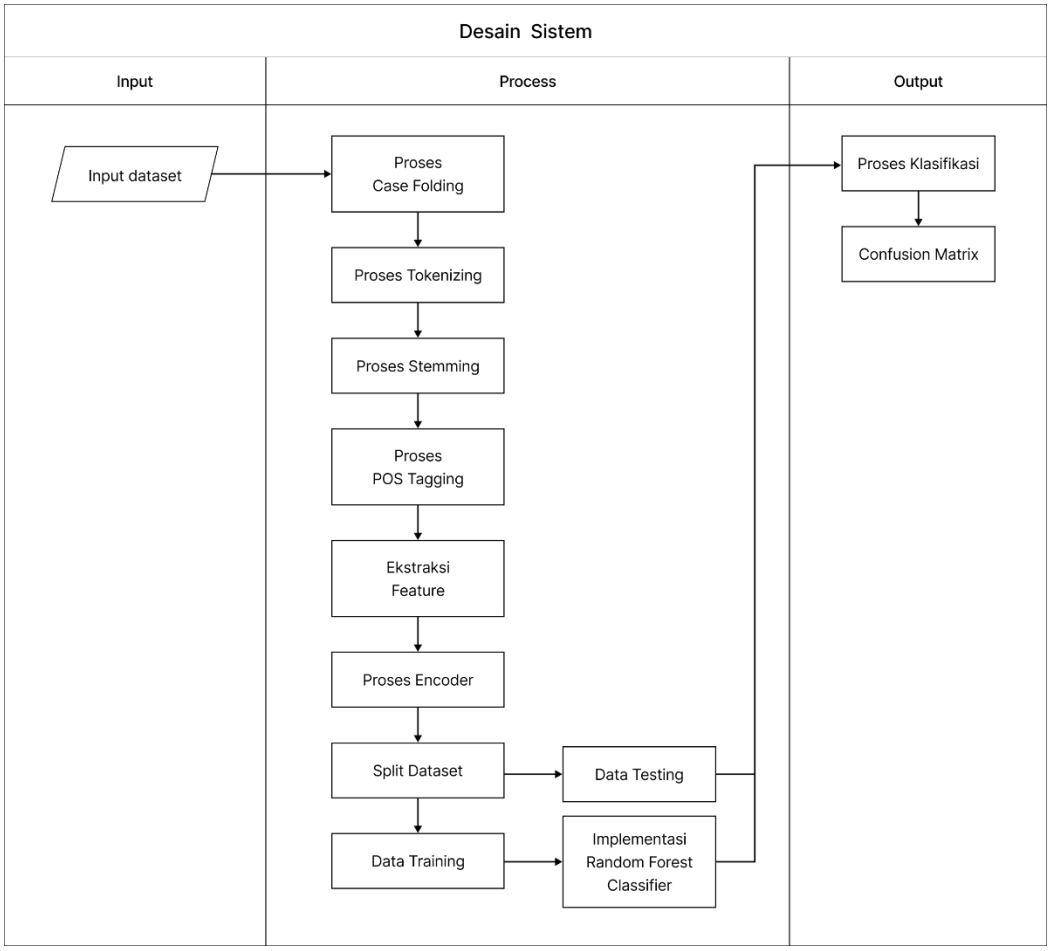
Dataset yang digunakan dalam penelitian berjumlah 5.238 kalimat percakapan. Data penelitian dibagi menjadi data latih dan data uji, data latih

nantinya akan diberikan 3 kelas label sesuai dengan kelas kalimat yaitu kelas pernyataan atau *statement* (S), pertanyaan atau *question* (Q), dan sapaan atau *chat* (C). Hasil dari pengumpulan data berupa *file* dengan tipe csv yang berisikan kalimat percakapan dan label dari kelas kalimat. Berikut contoh dataset yang yang terdapat dalam penelitian ini ditunjukkan oleh Tabel 3.1.

Tabel 3. 1 Contoh data teks dengan label

No	Kalimat	Kelas
1	Sorry I don't know about the weather.	S
2	what's the weather like today?	Q
3	I am fine	C
4	Where do you live?	Q
5	are you a chatbot?	Q
⋮	⋮	⋮
5234	Hello	C
5235	what are you doing?	Q
5236	one men cannot usually reproduce with each other	S
5237	a tree is a source of shelter / food for birds / animals in an ecosystem	S
5238	so good thanks	C

3.3 Perancangan Sistem



Gambar 3. 2 Perancangan Sistem

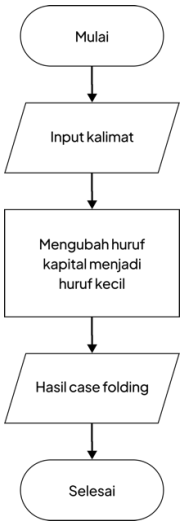
Berdasarkan gambar 3.2 diatas hal yang dilakukan pertama kali yakni input dataset kalimat percakapan. Dataset yang diinputkan terdiri dari dua variable yakni *sentences* dan *class*. Setelah dataset kalimat percakapan diinputkan maka data tersebut akan masuk ke proses *preprocessing*. Setelah *preprocessing*, data yang sudah diinputkan akan dilakukan proses ekstraksi fitur terlebih dahulu, setelah itu proses akan dilanjutkan kedalam proses *training* dengan menerapkan *Random Forest Classifier* dan proses *testing*. Setelah melalui tahap ini maka nantinya akan menghasilkan hasil klasifikasi.

3.3.1 *Preprocessing*

Preprocessing dilakukan untuk mendapatkan data yang berkualitas dan siap untuk diproses kedalam model. Pada tahap *preprocessing* menggunakan pendekatan *Natural Language Processing*. Tahapan *preprocessing* dalam penelitian ini sebagai berikut:

1. Proses *Case Folding*

Case Folding yaitu cara mengubah semua huruf yang terdapat dalam dokumen menjadi huruf kecil. Huruf yang digunakan adalah huruf ‘a’ sampai huruf ‘z’ dan karakter selain huruf tersebut dianggap sebagai pembatas. Pada proses *case folding* menggunakan fungsi *.lower()* untuk mengubah huruf menjadi kecil semua.



Gambar 3. 3 Alur *Case Folding*

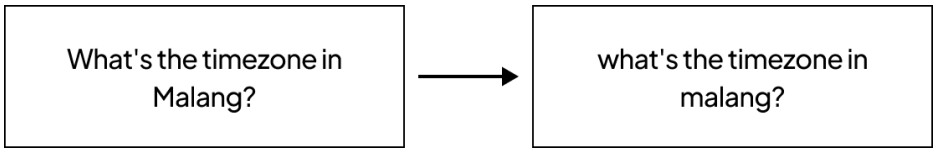
Keterangan :

a. Mulai.
b. Pengguna memasukkan kalimat yang akan diproses oleh program.
c. Setelah kalimat berhasil dimasukkan, program akan menjalankan fungsi *case folding* yang akan mengubah huruf besar atau kapital menjadi huruf kecil.

d. Output proses ini adalah mengubah huruf besar menjadi huruf kecil semua.

e. Selesai.

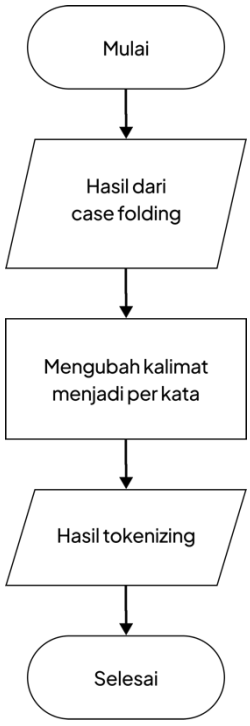
Pada Gambar 3.4. merupakan contoh dari proses *case folding* yang digunakan pada penelitian ini.



Gambar 3. 4 Contoh *Case Folding*

2. Proses *Tokenizing*

Tokenizing merupakan cara memisahkan kata masukan berupa kalimat menjadi perkata. Untuk memisahkan pertanyaan menjadi per kata menggunakan fungsi *.split()*.

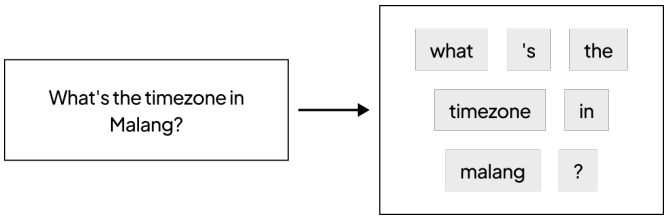


Gambar 3. 5 Alur *Tokenizing*

Keterangan :

a. Mulai
b. Hasil output proses case folding digunakan untuk proses *tokenizing* sebagai input.
c. Setelah hasil dari proses *case folding* digunakan sebagai input, selanjutnya kalimat akan diubah menjadi kata dengan menggunakan fungsi *.split()*.
d. Hasil dari proses ini adalah mengubah kalimat menjadi per kata.
e. Selesai

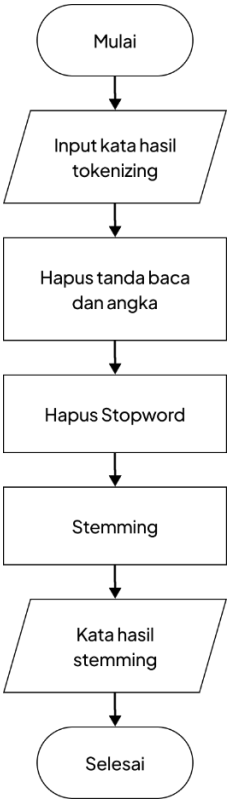
Setelah proses diatas selesai, Gambar 3.6 merupakan contoh dari hasil proses tokenizing pada penelitian ini.



Gambar 3. 6 Contoh *Tokenizing*

3. Stemming

Stemming merupakan cara menghilangkan imbuhan pada kata sehingga menghasilkan hanya kata dasar. Dalam satu kata dasar terdapat lebih dari satu imbuhan maka diperlukan urutan untuk menghilangkan kata imbuhan yang benar, jika salah dalam melakukan menghilangkan kata dasar maka kata dasar tersebut tidak dapat ditemukan. *Stemming* dapat digunakan untuk menemukan kata dasar dan menghasilkan struktur bahasa yang tepat.



Gambar 3. 7 Alur *Stemming*

Keterangan :

a. Mulai.

b. Hasil output proses *tokenizing* digunakan untuk proses *stemming* sebagai input.

c. Setelah output proses *tokenizing* digunakan sebagai input, selanjutnya dilakukan proses pembuangan tanda baca dan karakter spesial dari kalimat input.

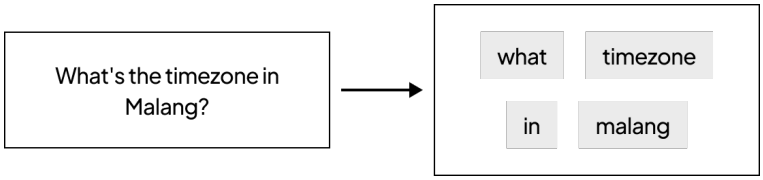
d. Setelah dilakukan proses pembuangan tanda baca, selanjutnya dilakukan proses hapus *stopword*. Proses ini dimaksudkan untuk mengetahui suatu kata masuk ke dalam *stopword* atau tidak. Proses ini dilakukan dengan pembuangan term yang tidak memiliki arti atau tidak relevan.

e. Setelah dilakukan proses pembuangan *stopword*, selanjutnya akan dilakukan proses *stemming*. Proses ini merupakan proses pembentukan kata dasar. *Stemming* digunakan untuk mengurangi bentuk term untuk menghindari ketidakcocokan yang dapat mengurangi *recall*, di mana *term* yang berbeda namun memiliki makna dasar yang sama direduksi menjadi satu bentuk. Proses *stemming* pada penelitian ini memanfaatkan modul NLTK dengan menggunakan algoritma *WordNetLemmatizer* dan *SnowBall*.

f. Output dari proses ini adalah mengubah kata hasil *tokenizing* ke dalam bentuk kata dasar.

g. Selesai.

Setelah proses diatas selesai, Gambar 3.8 merupakan contoh dari hasil proses stemming pada penelitian ini.



Gambar 3. 8 Contoh Stemming

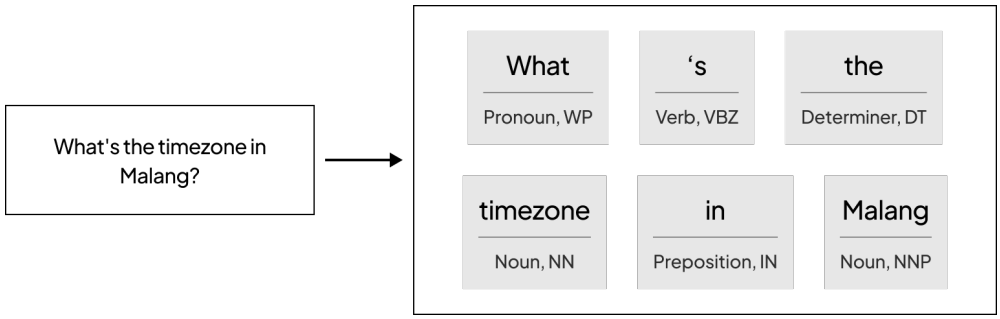
4. Proses *Part-of-Speech* (POS) *Tagging*

Part-of-Speech (POS) *tagging* atau secara singkat dapat ditulis sebagai *tagging* merupakan proses pemberian penanda *Part-of-Speech* atau kelas sintaktik pada setiap kata di dalam *corpus*. Dikarenakan *tag* secara umum juga diaplikasikan pada tanda baca, maka dalam proses *tagging*, tanda baca seperti tanda titik, tanda koma, dll perlu dipisahkan dari kata-kata.

Tag	Description	Example	Tag	Description	Example
CC	coord. conjunction	*and, or*	RB	adverb	*extremely*
CD	cardinal number	*one, two*	RBR	adverb, comparative	*never*
DT	determiner	*a, the*	RBS	adverb, superlative	*fastest*
EX	existential there	*there*	RP	particle	*up, off*
FW	foreign word	*noire*	SYM	symbol	*+, %*
IN	preposition or sub-conjunction	*of, in*	TO	"to"	*to*
JJ	adjective	*small*	UH	interjection	*oops, oh*
JJR	adject., comparative	*smaller*	VB	verb, base form	*fly*
JJS	adject., superlative	*smallest*	VBD	verb, past tense	*flew*
LS	list item marker	*1, one*	VBG	verb, gerund	*flying*
MD	modal	*can, could*	VBN	verb, past participle	*flown*
NN	noun, singular or mass	*dog*	VBP	verb, non-3sg pres	*fly*
NNS	noun, plural	*dogs*	VBZ	verb, 3sg pres	*flies*
NNP	proper noun, sing.	*London*	WDT	wh-determiner	*which, that*
NNPS	proper noun, plural	*Azores*	WP	wh-pronoun	*who, what*
PDT	predeterminer	*both, lot of*	WP$	possessive wh-	*whose*
POS	possessive ending	*'s*	WRB	wh-adverb	*where, how*
PRP	personal pronoun	*he, she*			

Gambar 3. 9 Penn Treebank Tagset

Fitur-fitur yang telah diekstraksi dari data untuk membangun model yang diperlukan dengan mengekstrak *Part-of-Speech tag* hingga menghasilkan fitur data numerik. Pada Gambar 3.10 merupakan contoh dari hasil proses POS *tagging* pada penelitian ini.



Gambar 3. 10 Contoh POS Tagging

5. Proses Ekstraksi *Feature*

Ekstraksi fitur merupakan proses mengubah teks mentah menjadi representasi numerik yang dapat dipahami oleh algoritma pembelajaran mesin atau

model statistik. Tujuan dari ekstraksi fitur adalah untuk mengonversi teks menjadi bentuk yang dapat digunakan untuk analisis lebih lanjut, seperti klasifikasi, klastering, atau pembuatan prediksi. Pada penelitian ini menggunakan ekstraksi TF (*Term Frequency*) untuk mengukur pentingnya sebuah kata dalam pembentukan pola sebuah kalimat. Ini menghasilkan skor numerik yang menggambarkan seberapa penting kata tersebut dalam pembentukan pola sebuah kalimat. Setelah proses ekstraksi fitur dilakukan, hasil yang didapatkan akan diekspor oleh sistem dalam bentuk format file *.csv*. Pada Gambar 3.11 merupakan contoh dari hasil proses ekstraksi fitur menggunakan ekstraksi TF (*Term Frequency*) pada penelitian ini.

Sentences	Class
but I cen't remember personal details	C

Hasil ekstraksi fitur			
wordCount	6	startTuple0	0
stemmedCount	5	endTuple0	0
stemmedEndNN	1	endTuple1	0
CD	0	endTuple2	0
NN	0	verbBeforeNoun	1
NNP	0	qMark	0
NNPS	0	qVerbCombo	1
PRP	1	qTripleScore	0
VBG	1	sTripleScore	0
VBZ	0	Class	C

Gambar 3. 11 Contoh Hasil Ekstraksi Fitur Menggunakan Ekstraksi TF

3.3.2 Pembagian *Dataset*

Dataset yang telah melalui proses ekstraksi fitur akan dilakukan pembagian atau *split dataset* dengan tujuan membagi *dataset* tersebut menjadi data *training* dan data *testing*. Data *training* merupakan data yang digunakan untuk melatih sebuah model. Sedangkan data *testing* merupakan data yang digunakan untuk proses testing atau validasi dari sebuah model (Mustafa et al., 2018). Dalam penelitian ini pembagian dataset dilakukan dengan rasio perbandingan 70:30 berdasarkan *rule of thumb* (aturan umum) secara acak.

3.3.3 Proses *Random Forest Classifier*

Dataset yang telah melalui proses ekstraksi fitur akan dilakukan pembagian atau *split dataset* dengan tujuan membagi *dataset* tersebut menjadi data *training* dan data *testing*. Data *training* merupakan data yang digunakan untuk melatih sebuah model. Sedangkan data *testing* merupakan data yang digunakan untuk proses testing atau validasi dari sebuah model (Mustafa et al., 2018). Dalam penelitian ini pembagian dataset dilakukan dengan rasio perbandingan 70:30 berdasarkan *rule of thumb* (aturan umum) secara acak.

Langkah-langkah dalam tahap perancangan penelitian *chatbot* "Gumara" pada telegram messenger dengan metode *Random Forest Classifier* yang dibuat adalah sebagai berikut:

1. Identifikasi masalah

   Pada tahapan identifikasi masalah merupakan langkah awal dalam proses pemecahan masalah berdasarkan latar belakang karena dengan

mengetahui dan memahami masalah yang ada, penulis dapat mengembangkan strategi dan solusi yang tepat untuk mengatasinya.

2. Studi literatur

Tahapan studi literatur dilakukan agar memperoleh referensi dan referensi sebanyak-banyaknya yang berkaitaan dengan lingkup pembahasan pada penelitian ini, dan juga mendapatkan informasi mengenai perkembangan metode yang digunakan dalam penelitian terkait. Studi literatur dan referensi yang digunakan adalah penelitian terdahulu yang membahas tentang *chatbot* dengan menggunakan metode yang berbeda dalam pemecahan masalahnya dan literatur tentang metode *Random Forest* dengan tujuan mendapatkan informasi-informasi yang dibutuhkan terkait dalam perancangan dan perkembangan *chatbot* pada telegram messenger dengan metode *Random Forest Classifier*, dan juga memberikan informasi yang terkait.

3. Pengumpulan data

Tahapan selanjutnya yaitu pengumpulan data. Pengumpulan data dilakukan untuk memperoleh informasi yang dibutuhkan dalam rangka mencapai tujuan penelitian. Pada tahapan pengumpulan data ini menggunakan data yang diperoleh dari situs Huging Face sebagai penyedia dataset percakapan kalimat yang sering dilakukan dalam Bahasa Inggris. Lalu data akan diolah hingga menghasilkan data fix yang akan digunakan sebagai *dataset* untuk proses *training* dan proses *testing*. Tahap berikutnya adalah tahapan desain sistem. Dalam sebuah penelitian,

tahapan desain sistem dilakukan agar proses penelitian dalam membuat *chatbot virtual route guide* dapat dilaksanakan secara terstruktur.

4. Desain sistem

   Dalam penelitian ini, tahapan desain sistem dilakukan agar proses penelitian dalam pembangunan *chatbot virtual route guide* dapat dilaksanakan secara terstruktur. Tahapan desain sistem dalam sebuah penelitian melibatkan beberapa langkah penting untuk memastikan bahwa sistem yang dikembangkan sesuai dengan kebutuhan dan tujuan penelitian.

5. *Preprocessing*

   Tujuan utama dari proses *preprocessing* menggunakan pendekatan NLP adalah untuk mempersiapkan data teks agar lebih mudah dipahami dan diolah oleh model atau algoritma *Random Forest Classifier*. Adapun tahap *preprocessing* yang digunakan pada penelitian ini sebagai berikut:

   a. *case folding* untuk mengubah semua huruf yang terdapat dalam dokumen menjadi huruf kecil,
   b. *tokenizing* untuk memisahkan kata masukan berupa kalimat menjadi perkata,
   c. *stemming* untuk untuk menemukan kata dasar dan menghasilkan struktur bahasa yang tepat,
   d. *POS tagging* untuk memberikan penanda *Part-of-Speech* atau kelas sintaktik pada setiap kata di dalam *corpus*.

6. Implementasi *Random Forest Classifier*

   Pada tahap ini dilakukan proses pembentukan model klasifikasi dengan menggunakan algoritma *Random Forest Classifier* dengan memanfaatkan konsep *bagging*. Beberapa pohon keputusan yang dibangun secara independen digabungkan untuk meningkatkan kinerja prediksi. Setiap pohon memberikan prediksi dan hasilnya digunakan dalam proses *voting* atau *averaging* untuk menentukan kelas prediksi akhir. Setelah model terbentuk lalu akan dilakukan proses pelatihan model menggunakan data pelatihan dari *dataset* untuk mengatur pohon-pohon keputusan dalam *ensemble*. Setiap pohon dibangun dengan menggunakan *subset* data pelatihan dan *subset* fitur yang dipilih secara acak dengan menggunakan kombinasi *hyperparameter* diantaranya adalah jumlah pohon (*n_estimator*), kedalaman pohon (*max_depth*), dan jumlah minimum sampel yang dibutuhkan untuk membagi sebuah *node* (*min_sample_split*) untuk meningkatkan kinerja model.

7. Pengujian sistem

   Tahapan ini bertujuan untuk menguji hasil model yang terlah dibentuk dari penerapan metode *Random Forest Classifier*. Melalui tahap pengujian sistem, akan dapat dipastikan apakah sistem yang dikembangkan dapat beroperasi secara efektif, dapat diandalkan, dan sesuai dengan tujuan penelitian yang telah ditetapkan.

8. Analisa hasil pengujian

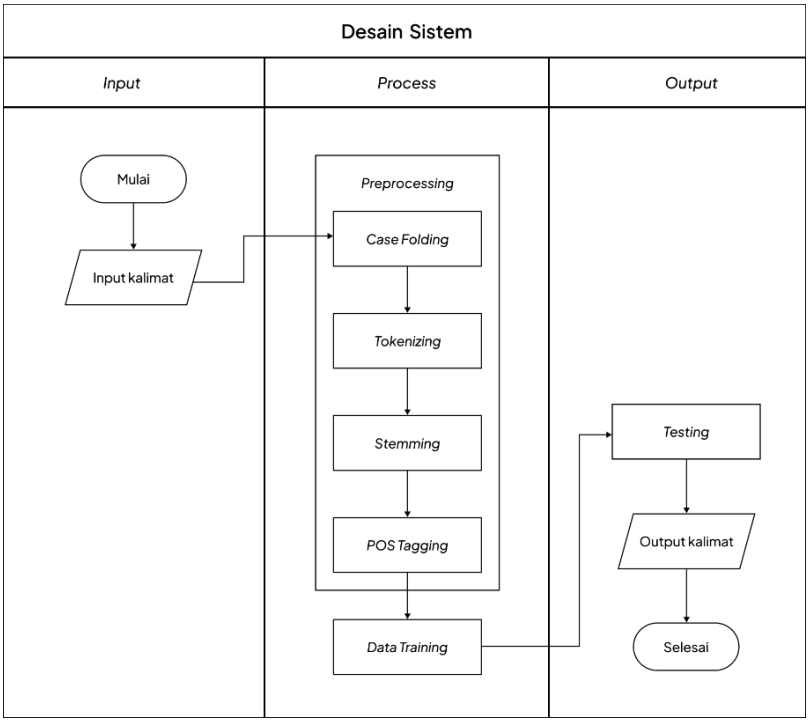
   Tahapan analisa hasil pengujian ini membantu memastikan bahwa hasil pengujian dievaluasi secara menyeluruh, masalah diidentifikasi dan ditangani, serta sistem siap untuk digunakan atau dikembangkan lebih lanjut sesuai kebutuhan penelitian. Hasil yang akan dianalisa dalam penelitian ini adalah apakah *chatbot virtual route guide* "Gumara" menggunakan metode *Random Forest Classifier* bisa digunakan atau tidak, dan seberapa besar hasil akurasi yang diperoleh *chatbot* dengan memanfaatkan pendekatan *Random Forest Classifier.*

9. Kesimpulan

   Pada tahapan kesimpulan ini dijelaskan tentang hasil rangkuman dari uji coba penelitian yang dilakukan tentang sistem yang dibuat. Dengan tahap kesimpulan, akan diketahui hasil dari penelitian yang telah dilakukan dan dapat diukur seberapa baik kualitas dari metode *Random Forest Classifier.*

3.4 Desain Sistem

Dalam sebuah penelitian dibutuhkan desain sistem agar tahapan penelitian dapat dilakukan secara terstruktur. Desain sistem dapat dilihat pada Gambar 3.2.



Gambar 3. 12 Desain Sistem

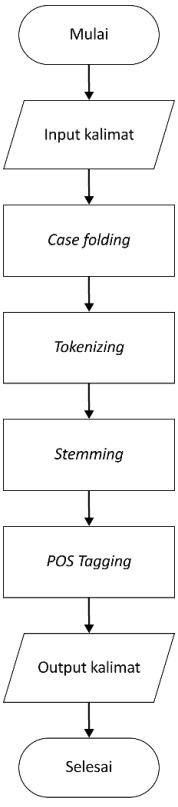
Pada tahap pertama terdapat *input* yang diberikan oleh pengguna berupa kalimat percakapan. Kemudian kalimat percakapan yang telah dimasukkan oleh pengguna akan memasuki tahap *process* yang terdapat tahapan proses *preprocessing* dan proses *training*. Pada tahap proses *preprocessing*, menggunakan pendekatan dari *Natural Language Processing* (NLP) yang terdiri dari *case folding* untuk mengubah semua huruf yang terdapat dalam dokumen menjadi huruf kecil, *tokenizing* untuk memisahkan kata masukan berupa kalimat menjadi perkata, *stemming* untuk untuk menemukan kata dasar dan menghasilkan struktur bahasa yang tepat, *POS tagging* untuk memberikan penanda *Part-of-Speech* atau kelas sintaktik pada setiap kata di dalam corpus.

Setelah proses *preprocessing* selesai, selanjutnya akan masuk ke dalam proses *data training* yaitu proses melatih data dari hasil *preprocessing*

menggunakan pendekatan *Random Forest Classifier*. Pada proses data training ini, data hasil preprocessing akan Pada tahap *data training* ini akan terbentuk sebuah model klasifikasi dari pelatihan dengan menggunakan metode *Random Forest Classifier*. Proses terakhir yaitu *testing* yang dilakukan untuk mendapatkan nilai akurasi dan *error* pada *chatbot*. Setelah proses selesai, maka sistem akan menghasilkan sebuah *output* kalimat yang akan dikirimkan sebagai kalimat balasan dari *input* kalimat kepada pengguna.

3.5 *Preprocessing*

Pada tahap *preprocessing* menggunakan pendekatan *Natural Language Processing*. Tahapan dalam *preprocessing* pada penelitian sebagai berikut:



Gambar 3. 13 *Flowchart Preprocessing*

Tahapan dari proses *preprocessing* yang digunakan pada penelitian ini adalah sebagai berikut:

1. *Case Folding*

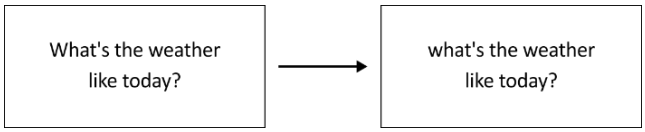
Case Folding yaitu cara mengubah semua huruf yang terdapat dalam dokumen menjadi huruf kecil. Huruf yang digunakan adalah huruf ‘a’ sampai huruf ‘z’ dan karakter selain huruf tersebut dianggap sebagai pembatas. Pada proses *case folding* menggunakan fungsi *.lower()* untuk mengubah huruf menjadi kecil semua.

Gambar 3. 14 Alur *Case Folding*

Keterangan :

a. Mulai.
b. Pengguna memasukkan kalimat yang akan diproses oleh program.
c. Setelah kalimat berhasil dimasukkan, program akan menjalankan fungsi *case folding* yang akan mengubah huruf besar atau kapital menjadi hirif kecil.
d. Output proses ini adalah mengubah huruf besar menjadi huruf kecil semua.
e. Selesai.

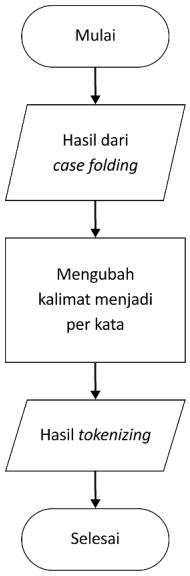
Pada Gambar merupakan contoh dari proses *case folding* yang digunakan pada penelitian ini.



Gambar 3. 15 Contoh *Case Folding*

2. *Tokenizing*

Tokenizing merupakan cara memisahkan kata masukan berupa kalimat menjadi perkata. Untuk memisahkan pertanyaan menjadi per kata menggunakan fungsi *.split()*.



Gambar 3. 16 Alur *Tokenizing*

Keterangan :

a. Mulai

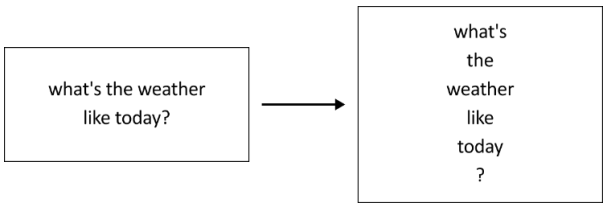
b. Hasil output proses *case folding* digunakan untuk proses *tokenizing* sebagai input.

c. Setelah output proses *case folding* digunakan sebagai input, selanjutnya kalimat akan diubah menjadi kata dengan menggunakan fungsi *.split()*.

d. Output dari proses ini adalah mengubah kalimat menjadi kata.

e. Selesai

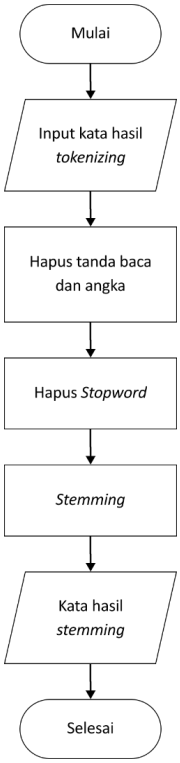
Setelah proses diatas selesai, Gambar 3.7 merupakan contoh dari hasil proses *tokenizing* pada penelitian ini.



Gambar 3. 17 Contoh *Tokenizing*

3. *Stemming*

Stemming merupakan cara menghilangkan imbuhan pada kata sehingga menghasilkan hanya kata dasar. Dalam satu kata dasar terdapat lebih dari satu imbuhan maka diperlukan urutan untuk menghilangkan kata imbuhan yang benar, jika salah dalam melakukan menghilangkan kata dasar maka kata dasar tersebut tidak dapat ditemukan. *Stemming* dapat digunakan untuk menemukan kata dasar dan menghasilkan struktur bahasa yang tepat.



Gambar 3. 18 Alur *Stemming*

Keterangan :

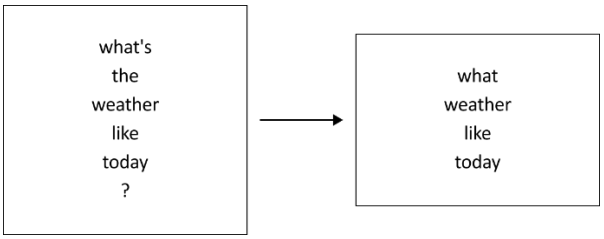
a. Mulai
b. Hasil output proses *tokenizing* digunakan untuk proses *stemming* sebagai input.
c. Setelah output proses *tokenizing* digunakan sebagai input, selanjutnya dilakukan proses pembuangan tanda baca dan karakter spesial dari kalimat input.
d. Setelah dilakukan proses pembuangan tanda baca, selanjutnya dilakukan proses hapus *stopword*. Proses ini dimaksudkan untuk mengetahui suatu kata masuk ke dalam *stopword* atau tidak. Proses ini dilakukan dengan pembuangan *term* yang tidak memiliki arti atau tidak relevan.
e. Setelah dilakukan proses pembuangan *stopword*, selanjutnya akan dilakukan proses *stemming*. Proses ini merupakan proses pembentukan kata dasar.

Stemming digunakan untuk mengurangi bentuk *term* untuk menghindari ketidakcocokan yang dapat mengurangi *recall*, di mana *term* yang berbeda namun memiliki makna dasar yang sama direduksi menjadi satu bentuk. Proses *stemming* pada penelitian ini memanfaatkan modul NLTK dengan menggunakan algoritma *WordNetLemmatizer* dan *SnowBall*.

f. Output dari proses ini adalah mengubah kata hasil tokenizing ke dalam bentuk kata dasar.

g. Selesai

Setelah proses diatas selesai, Gambar 3.9 merupakan contoh dari hasil proses *stemming* pada penelitian ini.



Gambar 3. 19 Contoh *Stemming*

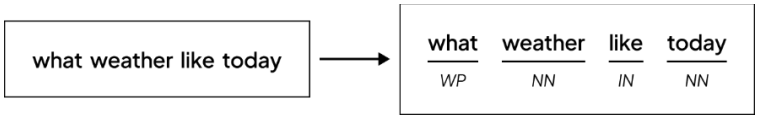
4. *Part-of-Speech Tagging* (*Tagging*)

Part-of-Speech (POS) *tagging* atau secara singkat dapat ditulis sebagai *tagging* merupakan proses pemberian penanda *Part-of-Speech* atau kelas sintaktik pada setiap kata di dalam corpus. Dikarenakan *tag* secara umum juga diaplikasikan pada tanda baca, maka dalam proses *tagging*, tanda baca seperti tanda titik, tanda koma, dll perlu dipisahkan dari kata-kata.

Tag	Description	Example	Tag	Description	Example
CC	coord. conjunction	*and, or*	RB	adverb	*extremely*
CD	cardinal number	*one, two*	RBR	adverb, comparative	*never*
DT	determiner	*a, the*	RBS	adverb, superlative	*fastest*
EX	existential there	*there*	RP	particle	*up, off*
FW	foreign word	*noire*	SYM	symbol	*+, %*
IN	preposition or sub-conjunction	*of, in*	TO	"to"	*to*
JJ	adjective	*small*	UH	interjection	*oops, oh*
JJR	adject., comparative	*smaller*	VB	verb, base form	*fly*
JJS	adject., superlative	*smallest*	VBD	verb, past tense	*flew*
LS	list item marker	*1, one*	VBG	verb, gerund	*flying*
MD	modal	*can, could*	VBN	verb, past participle	*flown*
NN	noun, singular or mass	*dog*	VBP	verb, non-3sg pres	*fly*
NNS	noun, plural	*dogs*	VBZ	verb, 3sg pres	*flies*
NNP	proper noun, sing.	*London*	WDT	wh-determiner	*which, that*
NNPS	proper noun, plural	*Azores*	WP	wh-pronoun	*who, what*
PDT	predeterminer	*both, lot of*	WP$	possessive wh-	*whose*
POS	possessive ending	*'s*	WRB	wh-adverb	*where, how*
PRP	personal pronoun	*he, she*			

Gambar 3. 20 *Penn Treebank Tagset*

Fitur-fitur yang telah diekstraksi dari data untuk membangun model yang diperlukan dengan mengekstrak *Part-of-Speech tag* hingga menghasilkan fitur data numerik. Pada Gambar 3.11 merupakan contoh dari hasil proses *POS tagging* pada penelitian ini.



Gambar 3. 21 Contoh *POS Tagging*

3.6 Implementasi *Random Forest Classifier*

3.6.1 Pembentukan Dasar *Decision Tree*

Tahap pertama dalam pembentukan *Decision Tree* adalah menghitung nilai ketidakpastian atau impurity dari dataset awal. Ukuran yang sering digunakan adalah *entropy* atau *Gini Index*. *Entropy* menggambarkan tingkat ketidakpastian distribusi kelas dalam data, sedangkan *Gini Index* menilai kemungkinan kesalahan

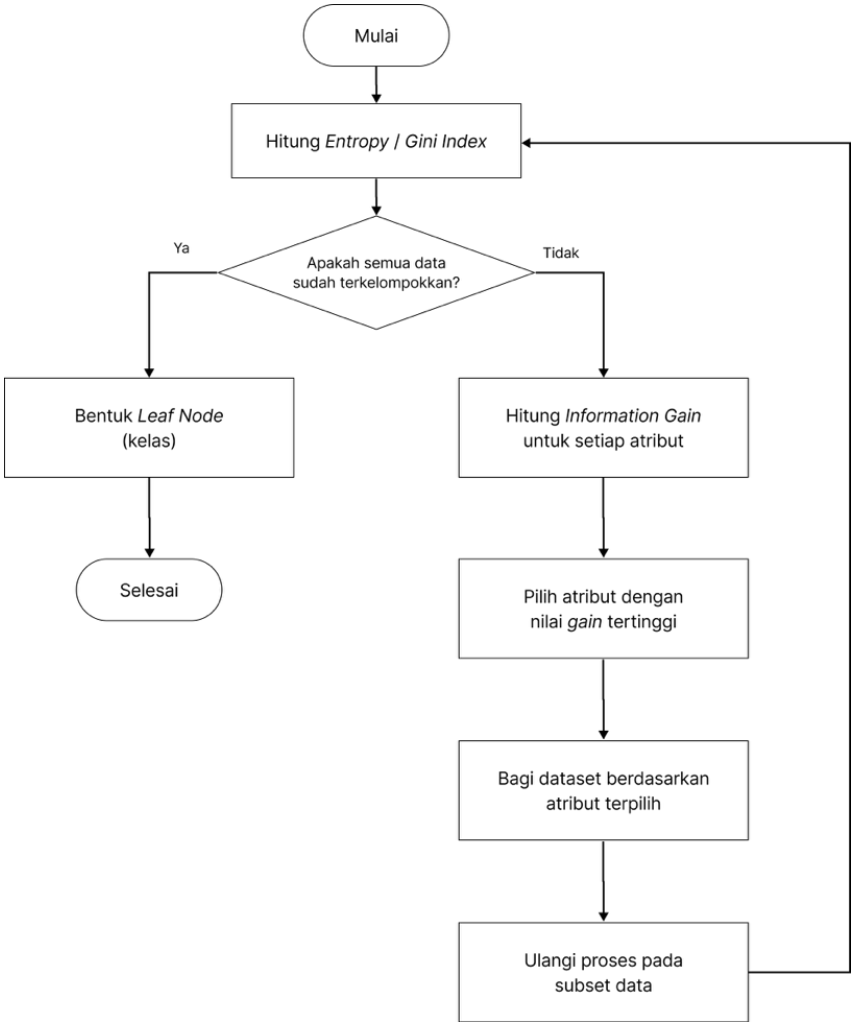
klasifikasi ketika data dipilih secara acak. Dengan mengukur nilai ini, dapat diketahui apakah data sudah terkelompokkan sesuai dengan atribut yang ditentukan atau data masih bercampur.

Setelah *entropy* dihitung, langkah selanjutnya adalah mencari nilai *information gain* dari setiap atribut yang tersedia. *Information gain* digunakan untuk mengukur seberapa besar pengurangan ketidakpastian setelah data dibagi berdasarkan suatu atribut tertentu. Atribut dengan nilai *information gain* tertinggi dipilih sebagai pemisah (*splitter*) pada *node* yang sedang diproses. Proses ini dilakukan secara berulang di setiap *node* sehingga terbentuk cabang-cabang baru yang membawa data semakin mendekati kelas target.

Proses pembentukan *node* dalam *Decision Tree* melibatkan pemilihan fitur dan menentukan titik potong (*cutting point*) untuk memisahkan data. Pemilihan *cutting point* dapat memengaruhi kualitas pohon yang terbentuk. Apabila *cutting point* dipilih dengan baik, maka data pada masing-masing cabang akan semakin terkelompokkan, sehingga memudahkan model dalam memberikan keputusan akhir yang akurat.

Pada setiap pembentukan *node*, algoritma juga melakukan evaluasi terhadap *candidate split* untuk melihat apakah pemisahan tersebut memberikan perbaikan signifikan terhadap pengelompokan data. Apabila *split* tidak memberikan perbaikan berarti, maka *node* tersebut akan dijadikan sebagai *leaf node* yang merupakan percabangan terakhir yang menyimpan keputusan klasifikasi berdasarkan mayoritas kelas dari data yang berada pada *node* tersebut.

Seiring dengan bertambahnya *node*, pohon akan terus berkembang hingga kondisi yang ditentukan tercapai. Beberapa kondisi penghentian yang umum digunakan antara lain adalah ketika semua data dalam node sudah terkelompokkan dan tidak ada lagi atribut yang tersisa untuk digunakan atau mencapai batas maksimal yang ditentukan. Kondisi penghentian ini penting agar pohon tidak tumbuh terlalu dalam yang dapat mengakibatkan *overfitting*. Proses pembuatan dasar *Decision Tree* dapat dilihat pada pada Gambar 3.22.



Gambar 3. 22 *Flowchart* Proses Pembentukan *Decision Tree*

3.6.2 Mekanisme *Bagging* pada *Random Forest*

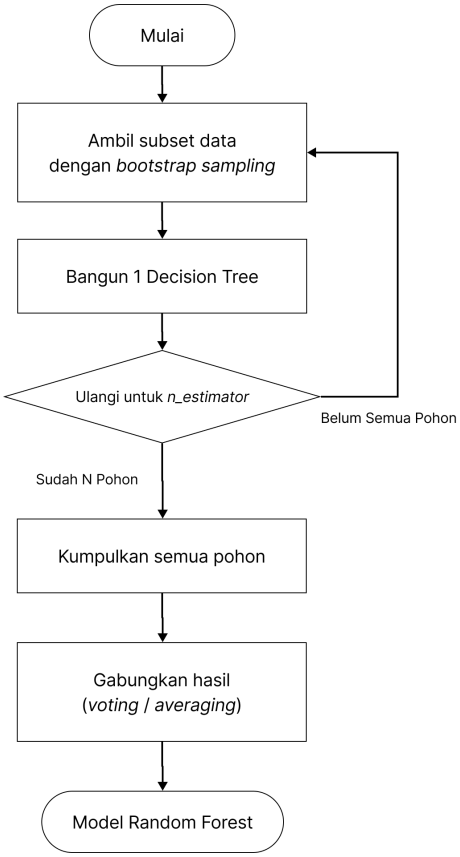
Bagging atau *Bootstrap Aggregating* merupakan inti dari metode *Random Forest*. Konsep ini muncul untuk mengatasi *overfitting* pada data latih. Dengan menggunakan *bagging*, banyak pohon keputusan dibangun dari subset data yang berbeda-beda. Setiap pohon dilatih secara independen sehingga hasil keseluruhan model lebih stabil dan akurat.

Proses *bagging* diawali dengan mengambil sampel secara acak dari dataset menggunakan teknik *bootstrap*. *Bootstrap* berarti pengambilan sampel dilakukan dengan pengembalian (*sampling with replacement*), sehingga beberapa data dapat muncul berulang kali dalam satu subset, sementara sebagian data lain tidak terambil. Cara ini membuat setiap pohon memiliki variasi input meskipun berasal dari dataset yang sama. Setelah sampel *bootstrap* terbentuk, setiap subset digunakan untuk melatih sebuah *Decision Tree*. Karena pohon-pohon ini dibangun dari data yang berbeda, struktur pohon yang dihasilkan juga bervariasi. Variasi ini justru bermanfaat karena menambah keragaman dalam prediksi dan mengurangi risiko semua pohon melakukan kesalahan yang sama.

Bagging memanfaatkan prinsip independensi. Pohon-pohon yang dilatih secara independen dengan data berbeda cenderung menghasilkan prediksi yang tidak saling bergantung. Akibatnya, ketika hasil prediksi digabungkan melalui voting atau rata-rata, kesalahan yang dilakukan oleh satu pohon dapat dikompensasi oleh pohon lain yang menghasilkan prediksi berbeda. Mekanisme *bagging* juga membantu dalam meningkatkan performa prediksi pada data baru. Dengan menggabungkan hasil dari banyak pohon, model dapat menangkap pola umum dari

data dan mengurangi pengaruh *outlier*. Inilah yang menjadikan *Random Forest* sering kali lebih unggul daripada satu *Decision Tree* tunggal. Jumlah pohon yang digunakan dalam *bagging* ditentukan oleh parameter *n_estimators*. Semakin banyak jumlah pohon, semakin stabil hasil prediksi, meskipun hal ini akan meningkatkan kebutuhan komputasi. Penelitian sebelumnya menunjukkan bahwa ratusan pohon biasanya sudah cukup untuk mendapatkan hasil akurasi yang optimal.

Selain variasi data, bagging juga memungkinkan variasi dalam pemilihan fitur. Pada *Random Forest*, setiap node pohon memilih subset fitur secara acak untuk membagi data. Hal ini meningkatkan diversifikasi model lebih jauh, karena setiap pohon bukan hanya berbeda dari sisi data, tetapi juga dari sisi fitur yang digunakan. Proses ini membuat *Random Forest* lebih tahan terhadap dominasi fitur tertentu. Jika dalam *Decision Tree* tunggal atribut tertentu selalu dipilih sebagai pemecah pertama, dalam *Random Forest* hal tersebut tidak selalu terjadi karena keterbatasan fitur yang dipilih secara acak pada setiap *node*. Mekanisme ini memastikan bahwa setiap pohon memiliki jalur pembelajaran yang unik. Tahapan mekanisme *bagging* pada *Random Forest* dapat dilihat pada Gambar 3.23.



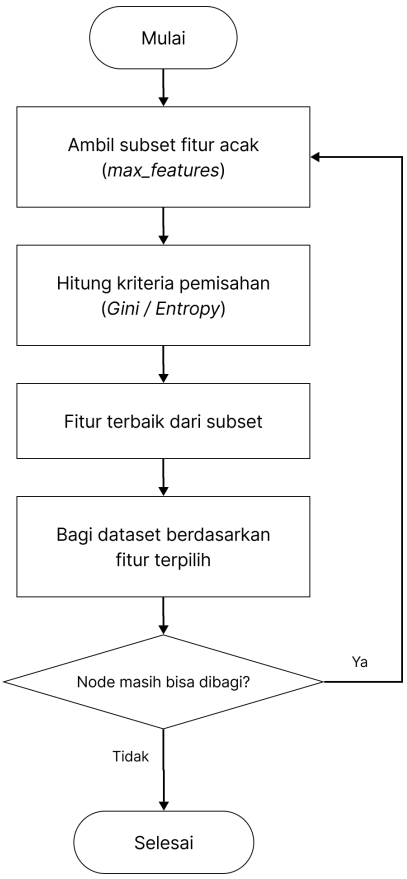
Gambar 3. 23 Flowchart Mekanisme *Bagging* Pada *Random Forest*

3.6.3 Pemilihan Fitur Secara Acak

Dalam proses ini, setiap node pada pohon keputusan tidak menggunakan seluruh fitur yang tersedia, melainkan hanya subset fitur yang dipilih secara acak. Proses pemilihan dimulai ketika algoritma akan membagi sebuah *node*. Algoritma secara acak akan memilih sejumlah fitur terbatas sesuai parameter *max_features* yang kemudian dipilih satu fitur terbaik berdasarkan kriteria pemisahan seperti *Gini Index* atau *Information Gain*. Parameter *max_features* mengatur jumlah fitur yang dipertimbangkan pada setiap *node*. Nilai parameter ini dapat berupa jumlah absolut, persentase, atau akar dari jumlah total fitur. Mekanisme ini memastikan setiap node bekerja dengan ruang pencarian terbatas.

Pengambilan fitur secara acak ini dilakukan pada setiap *node*. Ketika pohon berkembang, setiap cabang memiliki kemungkinan menggunakan kombinasi fitur yang berbeda. Hal ini membuat setiap jalur pohon merepresentasikan pola unik dari data yang mungkin tidak muncul pada pohon lain. Pemilihan fitur acak dilakukan dengan memanggil fungsi generator acak yang memilih indeks fitur dari seluruh ruang fitur. Indeks yang terpilih akan digunakan untuk menghitung nilai kriteria pemisahan. Proses ini berlangsung hingga semua *node* yang perlu dibagi selesai terbentuk.

Setelah subset fitur dipilih pada setiap *node*, algoritma menghitung kriteria pemisahan untuk masing-masing fitur dalam subset tersebut. Perhitungan ini meliputi evaluasi fitur yang memisahkan data ke dalam kelas target. Fitur dengan hasil pemisahan terbaik dipilih sebagai dasar percabangan *node* tersebut. Proses pemilihan ini diulang pada setiap cabang yang terbentuk. Dengan demikian, struktur pohon yang terbentuk akan berbeda-beda karena pemilihan subset fitur yang berbeda pada tiap node meskipun dataset dan label sama. Proses pemilihan fitur secara acak dapat dilihat pada pada Gambar 3.24.



Gambar 3. 24 *Flowchart* Pemilihan Fitur Secara Acak

3.6.4 Proses Voting *Random Forest*

Proses Voting dalam *Random Forest* merupakan tahap penting untuk menggabungkan prediksi yang dihasilkan oleh setiap *Decision Tree*. Pada tahap ini, setiap pohon yang terbentuk dari proses *bagging* dan pemilihan fitur acak akan memberikan hasil klasifikasi terhadap data uji. Hasil klasifikasi tersebut berupa label kelas yang dipilih pohon berdasarkan aturan yang terbentuk dari struktur *node* dan *leaf*. Seluruh label dari pohonn yang ada kemudian dikumpulkan untuk menentukan keputusan akhir. Konsep ini menyerupai pengambilan suara mayoritas, di mana kelas yang paling banyak dipilih dianggap sebagai hasil prediksi final.

Majority voting menjadi metode default dalam implementasi *Random Forest* untuk klasifikasi. Pada metode ini, setiap pohon memiliki bobot suara yang sama, sehingga tidak ada pohon yang lebih dominan daripada pohon lainnya. Misalnya, jika terdapat 200 pohon dan 120 di antaranya memprediksi kelas C sedangkan sisanya memprediksi kelas Q, maka hasil akhir adalah kelas C. Pemrosesan seperti ini memberikan hasil yang konsisten karena semua pohon diikutsertakan dalam jumlah yang seimbang. Dengan begitu, prediksi yang dihasilkan benar-benar merupakan representasi gabungan dari seluruh pohon.

Selain *majority voting*, parameter pendukung dalam proses voting adalah *n_estimators*. Parameter ini menentukan jumlah pohon yang dilibatkan dalam voting. Semakin banyak pohon yang digunakan, maka distribusi suara yang terbentuk untuk setiap data uji akan semakin lengkap juga. Jika jumlah pohon terlalu sedikit, hasil voting bisa menjadi kurang tepat karena variasi suara terbatas. Oleh karena itu, pemilihan jumlah pohon menjadi penting dalam memastikan hasil voting dapat menggambarkan pola dari seluruh dataset. Dengan *n_estimators* yang tepat, prediksi akhir akan menjadi lebih stabil karena suara mayoritas diperoleh dari representasi yang luas.

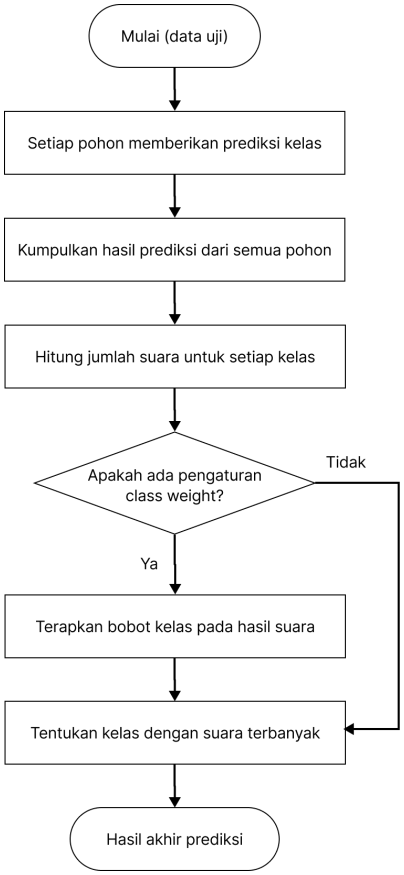
Parameter lain yang berhubungan dengan proses voting adalah *class_weight*. Parameter ini digunakan ketika distribusi kelas dalam dataset tidak seimbang. Dalam kondisi ini, tanpa pengaturan bobot yang tepat maka kelas yang dominan cenderung selalu menang dalam voting. Dengan memberikan bobot tertentu pada kelas minoritas, suara yang dihasilkan oleh pohon terhadap kelas tersebut bisa dipertimbangkan lebih besar. Hal ini membantu voting menghasilkan

keputusan yang lebih sesuai dengan distribusi kelas yang sebenarnya. Penerapan *class_weight* sangat relevan jika jumlah sampel antar kelas berbeda jauh. Dengan begitu, voting tetap dapat berjalan seimbang dalam kondisi data yang tidak proporsional.

Proses voting juga dipengaruhi oleh parameter *max_samples* ketika *bootstrap* digunakan. Parameter ini mengatur ukuran subset data yang diambil untuk melatih setiap pohon. Variasi ukuran subset akan menghasilkan pola prediksi yang sedikit berbeda antar pohon, yang kemudian mempengaruhi distribusi suara pada tahap voting. Dengan pengaturan *max_samples*, peneliti dapat mengontrol seberapa besar variasi suara antar pohon yang muncul. Hal ini memastikan bahwa suara mayoritas benar-benar mencerminkan hasil kolektif dari pohon yang dilatih dengan data berbeda. Dengan demikian, voting dipengaruhi tidak hanya oleh jumlah pohon, tetapi juga oleh variasi data yang melatih pohon.

Pada implementasi menggunakan library *scikit-learn*, hasil voting dapat diakses dengan parameter *predict_proba*. Fungsi ini memberikan distribusi kemungkinan dari suara yang diberikan oleh seluruh pohon terhadap setiap kelas. Meskipun keputusan akhir tetap diambil dari kelas dengan suara terbanyak, informasi kemungkinan ini dapat memberikan gambaran lebih rinci. Misalnya, jika kelas C memperoleh 70% suara dan kelas Q memperoleh 30%, maka meskipun hasil akhir adalah kelas C, pengguna tetap bisa mengetahui tingkat keyakinan model. Dengan demikian, probabilitas hasil voting dapat menjadi parameter tambahan dalam menganalisis keputusan *Random Forest*.

Secara keseluruhan, voting dalam *Random Forest* tidak hanya sekadar menghitung suara terbanyak, tetapi juga melibatkan pengaturan parameter yang dapat mempengaruhi distribusi suara. Parameter seperti *n_estimators*, *class_weight*, *max_samples*, dan fungsi *predict_proba* yang berperan dalam mengatur jalannya proses voting. Setiap pengaturan parameter akan menghasilkan pola suara yang sedikit berbeda, meskipun prinsip utama tetap sama yaitu memilih kelas dengan suara terbanyak. Dengan mekanisme ini, *Random Forest* dapat menghasilkan keputusan klasifikasi yang tepat dan konsisten. Proses voting dapat dilihat pada pada Gambar 3.25.



Gambar 3. 25 *Flowchart* Proses Voting *Random Forest* (Klasifikasi)

3.6.5 Pembentukan Model Klasifikasi *Random Forest*

Pembentukan model klasifikasi *Random Forest* dimulai dengan persiapan dataset yang dibagi menjadi data latih dan data uji. Data latih digunakan untuk melatih model, sedangkan data uji dipakai untuk mengevaluasi kinerjanya. Pada tahap ini, setiap *Decision Tree* dibangun dari subset data hasil *bootstrap* dan subset fitur yang dipilih secara acak. Seluruh proses ini dilakukan secara berulang untuk menghasilkan sejumlah pohon sesuai dengan nilai parameter *n_estimators*. Dengan cara tersebut, terbentuklah kumpulan pohon yang memiliki struktur dan aturan klasifikasi yang bervariari.

Parameter *n_estimators* berperan sebagai pengendali jumlah pohon dalam model. Semakin banyak jumlah pohon yang dibentuk, semakin lengkap pula kombinasi aturan klasifikasi yang terkandung dalam model. Nilai parameter ini biasanya ditentukan berdasarkan pengujian eksperimental untuk memperoleh performa yang stabil. Dalam penelitian tertentu, penggunaan ratusan pohon terbukti mampu menghasilkan beberapa prediksi yang konsisten. Oleh karena itu, pemilihan nilai *n_estimators* menjadi salah satu langkah krusial dalam membentuk model klasifikasi *Random Forest*. Setiap pohon yang terbentuk kemudian dilibatkan dalam proses voting akhir untuk menentukan label kelas.

Selain jumlah pohon, parameter *max_depth* juga penting dalam pembentukan model. Parameter ini menentukan kedalaman maksimum yang dapat dicapai oleh setiap pohon. Semakin besar nilai *max_depth*, maka pola yang bisa dipelajari akan semakin kompleks juga, tetapi akan menghasilkan resiko pembentukan struktur yang terlalu rumit. Apabila nilai kedalaman yang digunakan

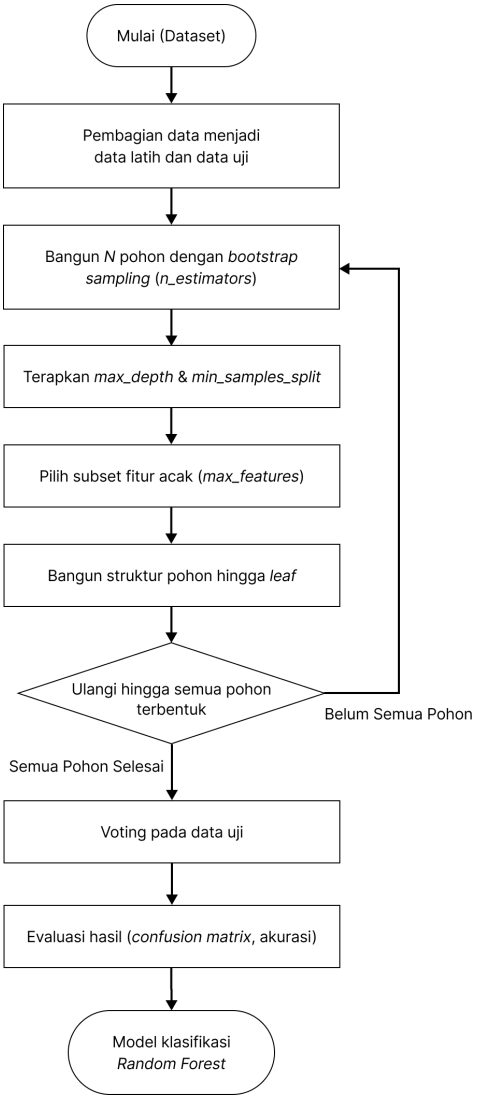
terbatas maka akan menghasilkan pohon dengan aturan yang lebih sederhana. Oleh karena itu, *max_depth* harus ditetapkan dengan pertimbangan yang matang agar setiap pohon tetap dapat menangkap pola penting tanpa menghasilkan percabangan yang berlebihan. Parameter ini secara langsung memengaruhi kualitas aturan klasifikasi pada level *node* hingga *leaf*.

Parameter lain yang berpengaruh adalah *min_samples_split*. Parameter ini menentukan jumlah minimum sampel yang dibutuhkan untuk membagi sebuah *node*. Jika jumlah sampel dalam suatu *node* lebih kecil dari nilai *min_samples_split*, maka *node* tersebut tidak akan dibagi lagi. Hal ini membatasi pertumbuhan pohon dan memastikan bahwa pembentukan aturan tidak dilakukan pada jumlah data yang terlalu kecil. Dengan pengaturan nilai ini, struktur pohon dapat dikendalikan agar lebih terarah dan tidak membentuk aturan yang terlalu spesifik.

Selain itu, parameter *max_features* digunakan untuk mengatur jumlah fitur yang dipertimbangkan pada setiap *node* ketika melakukan pemisahan. Dengan pengaturan parameter ini, setiap pohon akan memiliki variasi yang berbeda dalam pemilihan fitur yang digunakan untuk membentuk aturan klasifikasi. Variasi ini menjadi penting karena membuat jalur keputusan yang terbentuk pada tiap pohon menjadi unik. Semakin kecil nilai *max_features*, maka akan semakin besar keragaman antar pohon yang dihasilkan. Dengan begitu, parameter ini memberikan pengaruh yang signifikan terhadap pembentukan model yang beragam dalam *Random Forest*.

Setelah seluruh pohon selesai dibentuk, tahap selanjutnya adalah melakukan prediksi pada data uji menggunakan model yang telah dilatih. Setiap pohon

memberikan prediksi label untuk data uji, lalu hasilnya digabungkan melalui proses voting. Setelah proses tersebut dilakukan, maka kelas dengan jumlah suara terbanyak akan ditetapkan sebagai prediksi akhir. Prediksi ini kemudian dievaluasi menggunakan metrik klasifikasi seperti *confusion matrix*, akurasi, presisi, dan *recall*. Evaluasi ini memberikan gambaran mengenai seberapa baik akurasi dari model klasifikasi yang telah dibentuk. Proses pembentukan model klasifikasi *Random Forest* dapat dilihat pada pada Gambar 3.26.



Gambar 3. 26 *Flowchart* Pembentukan Model Klasifikasi *Random Forest*

BAB IV

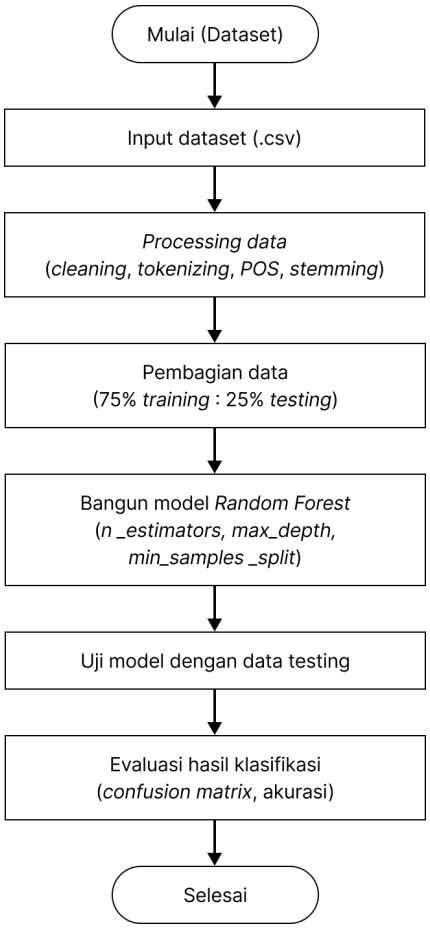
UJI COBA DAN PEMBAHASAN

4.1 Skenario Uji Coba

Bab ini menjelaskan secara rinci mengenai skenario uji coba yang dilakukan untuk mengukur kinerja sistem klasifikasi berbasis algoritma *Random Forest Classifier*. Uji coba dilakukan untuk memastikan bahwa sistem yang dibangun mampu mengolah data dengan baik mulai dari tahap masukan hingga keluaran hasil klasifikasi. Dalam penelitian ini, skenario uji coba disusun agar mencakup seluruh tahapan utama yang diperlukan dalam proses implementasi, sehingga memberikan gambaran menyeluruh mengenai performa sistem. Dengan adanya skenario yang jelas, pengujian dapat berjalan secara sistematis dan menghasilkan data evaluasi yang akurat. Selain itu, skenario uji coba ini juga memberikan dasar untuk membandingkan kinerja sistem pada berbagai variasi parameter model yang digunakan.

Skenario uji coba ini juga memperhatikan alur data dari awal hingga akhir agar setiap tahap saling terhubung dengan baik. Mulai dari input dataset, pembagian data, preprocessing, pembentukan model, hingga evaluasi, semua dilakukan secara berurutan. Dengan alur yang runtut, sistem dapat diuji secara konsisten dan hasil evaluasi dapat dibandingkan antar variasi parameter. Setiap tahap dalam skenario uji coba memiliki tujuan khusus yang saling mendukung. Tahap input dan *preprocessing* bertujuan memastikan kualitas data, tahap pembentukan model bertujuan menghasilkan struktur klasifikasi, tahap pengujian bertujuan menilai performa, dan tahap evaluasi bertujuan mengukur tingkat akurasi. Dengan tahapan

yang terstruktur, penelitian ini diharapkan dapat menghasilkan model klasifikasi yang akurat dan andal. Skenario uji coba dapat dilihat pada pada Gambar 4.1.



Gambar 4. 1 *Flowchart* Skenario Uji Coba

Flowchart skenario uji coba menggambarkan alur proses mulai dari input dataset hingga evaluasi hasil klasifikasi. Diagram ini memperlihatkan hubungan antar tahap sehingga dapat dipahami alur sistem secara visual.

4.1.1 Penggunaan Dataset

Dataset yang digunakan dalam penelitian ini terdiri dari kumpulan kalimat percakapan berbahasa Inggris yang diperoleh dari sumber *EU Open Data Portal* dan *HuggingFace* dengan tujuan menguji kemampuan sistem klasifikasi teks.

Dataset ini dipilih karena memiliki keragaman jenis kalimat yang mencakup pernyataan, pertanyaan, dan percakapan umum. Ketiga kategori ini mewakili tipe interaksi yang berbeda sehingga dapat dijadikan bahan uji yang baik untuk menguji performa dari algoritma *Random Forest Classifier*. Pemilihan dataset ini dilakukan dengan pertimbangan bahwa percakapan sehari-hari memiliki struktur yang beragam.

Secara keseluruhan, dataset berjumlah 5.238 kalimat percakapan yang telah melalui tahap pelabelan. Kalimat-kalimat ini dibagi ke dalam tiga kelas utama, yaitu *Statement* (S) sebanyak 1.473 data, *Question* (Q) sebanyak 1.238 data, dan *Chat* (C) sebanyak 3.073 data. Label yang diberikan pada setiap kalimat menunjukkan tipe komunikasi yang dimaksud sehingga memudahkan dalam proses klasifikasi. Proporsi dataset menunjukkan bahwa kelas *Chat* memiliki jumlah data terbanyak dibandingkan dua kelas lainnya. Dengan jumlah data yang cukup besar, dataset ini dianggap representatif untuk melatih dan menguji sistem. Format dataset disimpan dalam bentuk file (*.csv*) yang terdiri dari dua kolom utama. Kolom pertama berisi teks kalimat percakapan, sedangkan kolom kedua berisi label kategori (S, Q, atau C). Format ini dipilih karena mudah dibaca oleh perangkat lunak analisis data seperti *Python* dengan pustaka *pandas*. Selain itu, format (*.csv*) memungkinkan pengolahan lebih cepat dan sederhana. Struktur file yang rapi membantu mempermudah proses input data ke dalam sistem. Contoh sampel dari dataset yang digunakan pada penelitian ini ditunjukkan oleh tabel 4.1.

Tabel 4.1 Sampel dataset

No.	Komentar	Label
1	here is afternoon!	C
2	I'm fine too!! Happy to have someone to talk to	C
3	How are you today?	Q
4	Do you know else language?	Q
5	yellow animal are a mean of ruiner	S
6	a spreading is a mean of inanimate place	S

Proses pelabelan dataset dilakukan secara manual dengan meninjau makna setiap kalimat. Kalimat yang berbentuk pernyataan umum dikelompokkan sebagai *Statement*, kalimat yang diawali dengan kata tanya atau memiliki bentuk pertanyaan untuk mendapatkan informasi, konfirmasi atau penolakan terhadap sebuah pertanyaan ditempatkan pada kategori *Question*, sedangkan kalimat ringan sehari-hari yang bersifat interaktif masuk kategori *Chat*. Pelabelan ini penting agar akurasi label sesuai dengan konteks percakapan sebenarnya. Dengan metode ini, kejelasan data dapat terjamin sejak awal. Kualitas pelabelan menjadi salah satu faktor yang menentukan keberhasilan klasifikasi.

Dataset yang digunakan memiliki variasi panjang kalimat yang berbeda, mulai dari kalimat pendek hingga kalimat panjang dengan struktur kompleks. Hal ini memungkinkan sistem diuji pada berbagai tingkat kesulitan. Variasi ini diperlukan agar sistem mampu mengenali pola yang tidak terbatas pada satu bentuk kalimat. Dengan demikian, model yang dibangun lebih fleksibel dalam menghadapi data baru. Variasi panjang kalimat juga menguji kemampuan *preprocessing* dalam menormalisasi teks. Selain itu, dataset ini juga mengandung beragam kosakata,

termasuk kata formal dan informal. Kosakata formal banyak ditemukan pada kelas *Statement*, sedangkan kosakata informal sering muncul pada kelas *Chat*. Perbedaan kosakata ini memberikan karakteristik unik pada tiap kelas. Sistem diharapkan dapat mengenali kata-kata tertentu yang menjadi ciri khas masing-masing kelas.

Dalam tahap awal uji coba, dataset dibersihkan dari elemen-elemen yang tidak relevan seperti tanda baca berlebih, simbol khusus, dan karakter *non-alfabet.* Pembersihan ini dilakukan agar data lebih terstruktur dan siap diproses. Misalnya, tanda seru berlebih atau emotikon dihapus untuk menjaga konsistensi teks. Hanya kata-kata inti yang dipertahankan agar sistem dapat lebih fokus pada informasi penting. Pembersihan dataset merupakan tahap krusial sebelum dilakukan tokenisasi.

Dataset kemudian melalui proses tokenisasi, yaitu pemecahan kalimat menjadi kata-kata. Tokenisasi dilakukan agar sistem dapat menganalisis kata secara individual. Proses ini sangat membantu dalam membangun fitur yang akan digunakan oleh model klasifikasi. Dengan tokenisasi, panjang kalimat yang bervariasi dapat dinormalisasi dalam bentuk unit kata. Hasil tokenisasi juga memudahkan proses berikutnya, yaitu *POS tagging*.

Tahap berikutnya adalah *POS tagging* yang memberikan label kategori kata seperti *noun*, *verb*, atau *pronoun*. Label kategori kata ini memberikan informasi tambahan bagi model dalam membedakan kelas. Misalnya, keberadaan kata tanya seperti "*what*" atau "*how*" menjadi ciri khas kelas *Question*. Informasi tata bahasa ini menjadi salah satu fitur penting dalam model klasifikasi. Dengan *POS tagging*,

sistem tidak hanya mengenali kata, tetapi juga fungsi kata dalam kalimat. Proses ini meningkatkan kedalaman analisis teks.

Setelah *POS tagging*, dilakukan *stemming* dan *stopword removal*. *Stemming* berfungsi untuk mengubah kata ke bentuk dasarnya, sedangkan *stopword removal* menghapus kata-kata umum yang tidak memiliki makna yang terlalu berpengaruh seperti "*the*", "*is*", atau "*are*". Kedua tahap ini bertujuan memperbaiki kualitas dataset agar fitur yang digunakan lebih relevan. Dengan data yang lebih ringkas, sistem dapat bekerja lebih efisien. Penghapusan kata-kata umum juga membantu meningkatkan kejelasan pola antar kelas. Hasil *preprocessing* ini kemudian digunakan untuk membangun model.

4.1.2 Pembagian Dataset

Pembagian data merupakan langkah penting dalam skenario uji coba untuk memastikan model klasifikasi dapat diuji secara seimbang. Dataset yang digunakan dalam penelitian ini tidak langsung diproses seluruhnya, melainkan dibagi menjadi dua kategori data yaitu data latih (*training data*) dan data uji (*testing data*). Data latih digunakan untuk membentuk model klasifikasi, sementara data uji digunakan untuk mengukur sejauh mana model mampu menggeneralisasi pola baru. Dengan pembagian ini, peneliti dapat mengetahui performa model pada data yang tidak pernah dilihat sebelumnya. Tujuan utama dari pembagian data adalah mengurangi risiko *overfitting* yang terjadi jika model hanya diuji dengan data yang sama dengan data latih.

Dalam penelitian ini, rasio pembagian data ditetapkan sebesar 70% untuk data latih dan 30% untuk data uji. Rasio ini dipilih karena dianggap mampu memberikan

keseimbangan antara jumlah data yang cukup untuk melatih model dan jumlah data yang memadai untuk pengujian. Rasio 70:30 merupakan praktik umum yang banyak digunakan dalam penelitian berbasis NLP. (Bichri et al., 2024) dalam penelitian mereka yang berjudul "*Investigating the Impact of Train / Test Split Ratio on the Performance of Pre-Trained Models with Custom Datasets*" menegaskan bahwa proporsi ini memberikan keseimbangan yang baik antara jumlah data yang cukup untuk melatih model serta jumlah data yang representatif untuk mengevaluasi performa. Mereka juga menunjukkan bahwa rasio ini membantu menjaga konsistensi hasil eksperimen karena model diuji pada data yang belum pernah digunakan dalam proses pelatihan. (Vrigazova, 2021) dalam peneliannya yang berjudul "*The Proportion for Splitting Data into Training and Test Set for the Bootstrap in Classification Problems*" juga menekankan bahwa rasio 70:30 merupakan salah satu praktik umum dalam pembagian data klasifikasi. Menurutnya, proporsi ini sering digunakan karena mampu menjaga keseimbangan antara bias dan variansi dalam evaluasi model. Dengan 70% data latih, model memperoleh informasi yang memadai untuk mengenali pola, sementara 30% data uji cukup untuk memberikan gambaran nyata terhadap kemampuan generalisasi model. Dengan dasar tersebut, penelitian ini menetapkan rasio 70% data latih dan 30% data uji sebagai acuan utama dalam proses pembentukan model klasifikasi.

Dalam proses pembagian data dilakukan secara acak menggunakan fungsi bawaan dalam pustaka *scikit-learn*. Pengacakan tersebut dilakukan agar distribusi kelas dalam data latih dan data uji tetap seimbang. Misalnya, jika dalam dataset penuh terdapat 30% data kelas *Question*, maka proporsi yang sama juga diusahakan

ada dalam data latih maupun data uji. Dengan pengacakan ini, model dapat dilatih dan diuji pada data yang representatif dari keseluruhan dataset. Proses acak ini membantu menghindari bias yang mungkin terjadi jika data dibagi secara berurutan. Setiap kategori dalam dataset, yaitu *Statement*, *Question*, dan *Chat*, dijaga agar proporsinya tetap konsisten dalam pembagian. Hal ini disebut dengan *stratified sampling*, yang memastikan tidak ada kelas yang terlalu dominan dalam data latih atau data uji. Dengan cara ini, model dapat belajar dari semua kelas secara seimbang. Jika salah satu kelas terlalu sedikit muncul dalam data uji, hasil evaluasi bisa menjadi tidak akurat. Oleh karena itu, pembagian yang mempertahankan proporsi kelas menjadi penting untuk validitas hasil klasifikasi.

Data latih berfungsi sebagai bahan utama bagi model *Random Forest* untuk membangun aturan klasifikasi. Dari data latih ini, setiap *Decision Tree* dalam *Random Forest* dilatih untuk mengenali pola tertentu. Variasi data latih yang cukup besar memungkinkan terbentuknya banyak jalur keputusan. Dengan jumlah data latih yang memadai, model dapat membentuk pohon dengan struktur yang kuat. Semakin bervariasi pola dalam data latih, semakin kaya pula informasi yang dipelajari model. Data latih inilah yang menjadi pondasi dasar sistem klasifikasi. Sebaliknya, data uji digunakan untuk mengukur performa model yang sudah terbentuk. Data ini tidak pernah digunakan dalam proses pelatihan sehingga benar-benar menjadi data baru bagi model. Dengan menggunakan data uji, peneliti dapat menilai kemampuan generalisasi model. Jika akurasi tinggi pada data latih tetapi rendah pada data uji, maka model dianggap mengalami *overfitting*. Sebaliknya, jika akurasi konsisten pada kedua jenis data, berarti model memiliki hasil yang baik.

4.1.3 Proses Pembentukan Model Klasifikasi

Proses klasifikasi dalam penelitian ini diawali dengan pembentukan model menggunakan algoritma *Random Forest Classifier*. Model ini dibangun berdasarkan dataset yang telah melalui tahap *preprocessing*. *Preprocessing* meliputi pembersihan teks, tokenisasi, dan penghapusan *stopwords* agar data menjadi lebih terstruktur. Setelah tahap ini, sistem dilengkapi dengan informasi tambahan berupa *POS tagging* untuk setiap kata dalam kalimat. *POS tagging* ini kemudian digunakan sebagai salah satu dasar dalam pembentukan aturan klasifikasi. Dengan cara ini, sistem dapat mengenali bukan hanya kata, tetapi juga fungsi kata dalam kalimat.

POS tagging atau *part-of-speech tagging* adalah proses memberikan label kategori kata pada setiap token hasil tokenisasi. Label yang digunakan meliputi kategori seperti *noun* (NN), *proper noun* (NNP), *plural noun* (NNPS), *cardinal number* (CD), *pronoun* (PRP), *verb* (VB), *adjective* (JJ), dan sebagainya. Label-label ini memberikan informasi tambahan mengenai struktur sintaksis kalimat. Dengan mengetahui kategori kata, sistem dapat lebih mudah mengidentifikasi pola khas dari suatu kalimat. Sebagai contoh, kalimat yang berbentuk pertanyaan biasanya diawali dengan kata tanya (*wh-word*) seperti "*what*", "*who*", atau "*how*". Kata-kata ini diberi label tertentu dalam *POS tagging*, misalnya "*WP*" untuk *who/what* dan "*WRB*" untuk *how*. Dengan mengenali keberadaan label-label ini, sistem dapat lebih cepat mengenali pola pertanyaan. Sebaliknya, kalimat deklaratif atau *statement* umumnya lebih banyak mengandung kata benda (NN, NNP) atau

angka (CD) yang menjelaskan objek tertentu. Sementara itu, kalimat *chat* lebih cenderung sederhana dan mengandung kata-kata informal yang sering diulang.

Proses klasifikasi melibatkan pengaturan parameter utama dalam *Random Forest*. Jumlah pohon (*n_estimators*), kedalaman pohon (*max_depth*), dan jumlah minimum sampel split (*min_samples_split*) digunakan untuk mengatur kompleksitas model. Dengan data yang diperkaya informasi *POS tagging*, setiap pohon akan membentuk aturan berdasarkan gabungan kata dan label kategori kata. Contohnya, aturan pohon dapat berupa "Jika kalimat mengandung label WP atau WRB, maka lebih cenderung termasuk kategori *Question*". Dengan banyak pohon, model akan membentuk beragam aturan serupa. Kombinasi aturan inilah yang menghasilkan keputusan akhir. Selain itu, distribusi label POS dalam dataset juga diperhitungkan. Misalnya, kategori *Statement* memiliki lebih banyak kata benda (NN, NNP, NNPS), kategori *Question* banyak mengandung kata tanya (WP, WRB), sedangkan kategori *Chat* memiliki kombinasi kata sederhana dengan struktur lebih bebas. Distribusi ini digunakan oleh model untuk mengenali perbedaan pola antar kategori.

Tahap berikutnya adalah melatih model dengan data latih yang sudah dilabeli. Pada tahap ini, algoritma *Random Forest* membentuk aturan klasifikasi dengan membagi data berdasarkan fitur yang paling informatif. Setiap pohon dilatih secara independen dengan subset data dan subset fitur yang berbeda. Hasilnya adalah kumpulan pohon dengan aturan yang bervariasi, termasuk aturan berbasis *POS tagging*. Keragaman aturan ini memberikan dasar bagi model untuk melakukan klasifikasi dengan hasil yang konsisten. Setelah model selesai dilatih,

data uji dimasukkan ke dalam sistem untuk dilakukan prediksi. Data uji juga melalui *preprocessing* dan *POS tagging* yang sama dengan data latih. Setiap kalimat dalam data uji dianalisis berdasarkan aturan yang telah dibentuk oleh pohon-pohon dalam *Random Forest*. Setiap pohon memberikan prediksi label untuk kalimat tersebut. Kemudian hasil prediksi digabungkan melalui proses voting mayoritas untuk menentukan kelas akhir. Proses ini memastikan bahwa semua pohon berkontribusi dalam keputusan akhir.

Hasil klasifikasi kemudian dievaluasi untuk melihat sejauh mana model mampu mengenali pola kalimat berdasarkan kombinasi kata dan *POS tagging*. Evaluasi dilakukan menggunakan *confusion matrix* yang menunjukkan jumlah prediksi benar dan salah untuk setiap kategori. Dari *confusion matrix*, nilai akurasi dapat dihitung sebagai indikator utama performa sistem. Analisis hasil evaluasi ini menunjukkan efektivitas penggunaan *POS tagging* dalam memperbaiki kualitas klasifikasi. Dengan demikian, proses klasifikasi menghasilkan data kuantitatif yang dapat diinterpretasikan. Proses klasifikasi juga memperlihatkan hubungan antara parameter model dengan kinerja hasil klasifikasi. Misalnya, peningkatan jumlah pohon dapat memperkuat stabilitas hasil, sementara kedalaman pohon memengaruhi kompleksitas aturan yang terbentuk. Jumlah minimum sampel split mengatur keseimbangan dalam pembentukan aturan berbasis *POS tagging*. Kombinasi parameter yang berbeda memberikan hasil akurasi yang bervariasi. Dengan menganalisis hasil dari berbagai kombinasi parameter, peneliti dapat menentukan konfigurasi terbaik. Analisis ini dilakukan secara menyeluruh untuk setiap kategori data.

4.2 Implementasi Sistem

Pada bagian ini dibahas secara rinci bagaimana sistem klasifikasi berbasis algoritma *Random Forest Classifier* diimplementasikan. Implementasi sistem dimaksudkan untuk menerjemahkan rancangan konseptual yang telah dibahas pada bab sebelumnya menjadi langkah teknis yang dapat dijalankan pada perangkat lunak. Dengan adanya implementasi ini, proses klasifikasi dapat dilakukan secara terstruktur mulai dari tahap inisialisasi data, pembentukan model klasifikasi, hingga pengujian sistem.

4.2.1 Proses Inisialisasi Data

Tahap inisialisasi data merupakan langkah awal yang sangat penting dalam implementasi sistem. Pada tahap ini, dataset yang digunakan dimasukkan ke dalam sistem dalam format yang sesuai untuk dianalisis. Dataset yang digunakan dalam penelitian ini berbentuk file *.csv* dengan struktur dua kolom utama, yaitu kolom teks percakapan dan kolom label kategori. Label kategori terdiri dari tiga kelas utama: *Statement* (S), *Question* (Q), dan *Chat* (C). Format sederhana ini dipilih agar mudah diproses oleh perangkat lunak *Python*, khususnya melalui pustaka *pandas* yang mendukung manipulasi data yang disajikan dalam format tabel. Pada tahap ini, dilakukan pemeriksaan awal untuk memastikan tidak ada nilai kosong atau baris yang rusak dalam file dataset yang kemudian dilanjutkan dengan insialisasi daftar kata kunci dan aturan untuk fitur-fitur yang digunakan dalam proses klasifikasi. Inisialisasi daftar kata kunci membantu memperkuat fitur teks yang paling sering muncul dan relevan dengan kategori tertentu. Penerapan proses inisialisasi ditunjukkan pada Gambar 4.2.

```
startTuples = ["NNS-DT", "WP-VBZ", "WRB-MD"]

endTuples = ["IN-NN", "VB-VBN", "VBZ-NNP"]

feature_keys = ["id", "wordCount", "stemmedCount", "stemmedEndNN", "CD", "NN",
                "NNP", "NNPS", "NNS", "PRP", "VBG", "VBZ", "startTuple0",
                "endTuple0", "endTuple1", "endTuple2", "verbBeforeNoun",
                "qMark", "qVerbCombo", "qTripleScore", "sTripleScore", "class"]
```

Gambar 4. 2 Penerapan Kata Kunci dan Aturan Fitur

Selanjutnya, dilakukan tahap pembersihan teks sebagai bagian dari inisialisasi data. Pembersihan ini bertujuan untuk menghilangkan karakter-karakter yang tidak relevan seperti tanda baca, simbol khusus, angka yang tidak berhubungan dengan konteks, serta spasi berlebih. Hanya kata-kata yang memiliki makna signifikan yang dipertahankan. Dengan langkah ini, teks menjadi lebih bersih dan seragam untuk diproses pada tahap selanjutnya. Proses ini mempermudah sistem untuk fokus pada kata, frasa, atau kalimat yang penting dalam proses klasifikasi. Penerapan proses pembersihan karakter pada teks ditunjukkan pada Gambar 4.3.

```
def strip_sentence(sentence):
    sentence = sentence.strip(",")
    sentence = "".join(filter(lambda x: x in string.printable, sentence))
    sentence = sentence.translate(str.maketrans("", "", string.punctuation))
    return sentence
```

Gambar 4. 3 Penerapan Proses Pembersihan Karakter pada Teks

Setelah tahap pembersihan data selesai, teks yang diperoleh dari dataset kemudian diproses melalui tahapan tokenisasi dan *POS tagging*. Implementasi tokenisasi dilakukan dengan menggunakan pustaka *NLP* seperti *NLTK* dan *spaCy* yang menyediakan fungsi bawaan untuk memecah kalimat menjadi token. Pada penelitian ini, fungsi *word_tokenize()* dari *NLTK* digunakan untuk membagi setiap kalimat ke dalam potongan kata. Hasil tokenisasi ini kemudian ditampilkan dalam

bentuk list agar dapat diproses pada tahap selanjutnya. Dengan cara ini, setiap baris kalimat yang panjang dapat diubah menjadi kumpulan kata individu. Sedangkan pada *POS tagging* memberikan label gramatikal pada setiap token, misalnya NN untuk *noun*, NNP untuk *proper noun*, VB untuk *verb*, dan CD untuk *cardinal number*. Dengan adanya label ini, sistem tidak hanya mengenali kata, tetapi juga fungsi kata dalam kalimat. Informasi tambahan ini sangat membantu dalam membedakan pola kalimat antara *Statement*, *Question*, dan *Chat*. Misalnya, kalimat tanya cenderung mengandung label WP atau WRB, sementara kalimat pernyataan lebih sering mengandung NN dan NNP. Penerapan proses tokenisasi dan *POS tagging* ditunjukkan pada Gambar 4.4.

```
def get_pos(sentence):
    sentenceParsed = word_tokenize(sentence)
    return nltk.pos_tag(sentenceParsed)
```

Gambar 4. 4 Penerapan Proses Tokenisasi dan *POS Tagging*

Selain *POS tagging*, tahap inisialisasi juga mencakup *stemming* dan penghapusan *stopwords*. *Stemming* bertujuan mengubah kata menjadi bentuk dasarnya, sedangkan *stopword removal* menghapus kata-kata umum seperti “the”, “is”, atau “are” yang tidak memberikan kontribusi signifikan terhadap klasifikasi. Dengan langkah ini, teks menjadi lebih ringkas tanpa kehilangan makna penting. Proses ini membantu mengurangi kompleksitas data dan meningkatkan fokus model pada kata-kata kunci yang relevan. Hasil *stemming* dan penghapusan *stopwords* menghasilkan representasi teks yang lebih optimal untuk analisis. Langkah ini menjadikan data lebih efisien untuk diolah pada tahap pembentukan

model. Penerapan proses *stemming* dan penghapusan *stopwords* ditunjukkan pada Gambar 4.5.

```
def stematize(sentence):

    stop_words = set(stopwords.words("english"))
    word_tokens = word_tokenize(sentence)

    filtered_sentence = []
    for w in word_tokens:
        if w not in stop_words:
            filtered_sentence.append(w)
    stemmed = []
    for w in filtered_sentence:
        stemmed.append(sno.stem(w))

    return stemmed
```

Gambar 4. 5 Penerapan Proses *Stemming* dan *Stopword Removal*

Dalam tahap inisialisasi, dilakukan pula pengkodean label kelas menjadi format numerik agar mudah diproses oleh algoritma. Label kategori seperti S, Q, dan C diubah menjadi angka yang mewakili masing-masing kelas. Proses ini diperlukan karena algoritma klasifikasi bekerja dengan data numerik, bukan teks mentah. Dengan konversi ini, sistem dapat mengolah label kelas secara lebih efisien dalam proses pelatihan dan pengujian. Pengkodean label juga mempermudah perhitungan metrik evaluasi seperti *confusion matrix*. Selain itu, inisialisasi data mencakup penyimpanan ulang data yang telah melalui *preprocessing* ke dalam file baru. File ini disimpan sebagai file dalam format *.csv* untuk memastikan bahwa data yang digunakan dalam proses pelatihan dan pengujian konsisten. Penyimpanan data yang sudah diproses juga bermanfaat jika penelitian perlu diulang atau dibandingkan dengan metode lain. Dengan demikian, proses eksperimen dapat direplikasi dengan hasil yang sama.

4.2.2 Pembentukan Model Klasifikasi

Proses pembentukan model klasifikasi dimulai setelah dataset berhasil melalui tahap inisialisasi. Dataset yang telah dibersihkan, ditokenisasi, diberi label *POS tagging*, serta dibagi menjadi data latih dan data uji menjadi masukan utama dalam proses ini. Tujuan utama dari pembentukan model adalah membangun struktur klasifikasi yang mampu mengenali pola kalimat berdasarkan data latih yang tersedia. Dengan memanfaatkan algoritma *Random Forest Classifier*, proses ini dirancang agar sistem dapat menghasilkan keputusan klasifikasi yang akurat. *Random Forest* dipilih karena kemampuannya menggabungkan banyak pohon keputusan sehingga memperkuat hasil prediksi.

Tahap pertama dalam pembentukan model adalah memastikan dataset telah terbagi sesuai dengan rasio yang telah ditentukan, yaitu 70% data latih dan 30% data uji. Pembagian ini dilakukan dengan menggunakan fungsi *train_test_split* dari pustaka *scikit-learn* yang mendukung pembagian data secara acak dengan mempertahankan distribusi kelas. Dengan cara ini, setiap kelas tetap memiliki representasi yang seimbang dalam data latih maupun data uji. Data latih digunakan untuk membentuk pola pada model, sedangkan data uji digunakan untuk mengevaluasi hasil. Tanpa adanya pembagian ini, model tidak dapat diuji secara objektif. Oleh karena itu, tahap pembagian dataset menjadi fondasi awal dalam pembentukan model klasifikasi.

Setelah dataset terbagi, langkah berikutnya adalah menentukan parameter utama yang akan digunakan dalam *Random Forest*. Parameter yang paling penting adalah jumlah pohon atau *n_estimators*, yang menentukan berapa banyak pohon

keputusan akan dibentuk dalam model. Semakin banyak jumlah pohon yang dibentuk, semakin beragam pula aturan klasifikasi yang dihasilkan. Dalam penelitian ini, nilai *n_estimators* diatur pada beberapa variasi, yaitu 100, 200, dan 500, agar dapat dibandingkan performanya. Variasi ini memungkinkan peneliti untuk mengetahui dampak jumlah pohon terhadap tingkat akurasi. (Stefan Wager, n.d.) dalam jurnal yang berjudul "*Asymptotic Theory for Random Forests*" menjelaskan bahwa peningkatan jumlah pohon pada *Random Forest* akan menurunkan varian prediksi dan membuat hasil klasifikasi lebih stabil. Namun, setelah jumlah pohon mencapai titik tertentu, penambahan pohon berikutnya hanya memberikan peningkatan tambahan terhadap performa. Oleh karena itu, penggunaan jumlah pohon seperti 100, 200, hingga 500 pada penelitian ini dapat dianggap cukup representatif untuk mencapai kestabilan model tanpa menambah beban komputasi berlebih.

Selain *n_estimators*, parameter lain yang sangat berpengaruh adalah *max_depth*. Parameter ini mengatur kedalaman maksimum dari pohon keputusan yang dibentuk dalam *Random Forest*. Jika nilai *max_depth* terlalu kecil, maka pohon yang terbentuk tidak dapat menangkap pola kompleks dari data. Sebaliknya, jika nilainya terlalu besar, pohon bisa menjadi terlalu rumit dan berisiko memngalammi *overfitting*. Oleh karena itu, penelitian ini mengatur nilai *max_depth* dengan variasi 5, 10, dan 20. Dalam penelitian (Sun et al., 2024), penulis menggunakan pendekatan *grid search* untuk memilih parameter *Random Forest*, termasuk batas kedalaman pohon (*max_depth*). Mereka menyebut bahwa pengaturan kedalaman maksimum sangat penting untuk menyeimbangkan

kemampuan generalisasi dan kompleksitas model agar tidak *overfitting*. Dengan demikian, penggunaan nilai max_depth seperti 5, 10, atau 20 dalam penelitian ini didasarkan pada praktik optimasi parameter yang juga diadopsi dalam studi tersebut.

Parameter berikutnya adalah *min_samples_split*, yaitu jumlah minimum sampel yang diperlukan untuk membagi suatu *node*. Jika nilai parameter ini terlalu kecil, pohon dapat membentuk aturan dari jumlah data yang sangat sedikit sehingga hasilnya kurang stabil. Jika nilainya terlalu besar, pohon akan menjadi terlalu sederhana dan gagal menangkap variasi data. Penelitian ini menggunakan variasi parameter yaitu 2, 5, dan 10. (Patel & Giri, 2016) dalam penelitiannya yang berjudul "Feature Selection and Classification of Mechanical Fault of An Induction Motor Using Random Forest Classifier" menegaskan bahwa pemilihan nilai parameter *min_samples_split* dalam *Random Forest* berperan penting untuk mengontrol pembentukan pohon. Mereka menunjukkan bahwa nilai kecil, seperti 2 atau 5, memungkinkan pohon tumbuh lebih rinci, sedangkan nilai yang lebih besar seperti 10 membantu mencegah *overfitting* dengan membatasi pembagian *node*. Oleh karena itu, variasi nilai 2, 5, dan 10 layak diuji untuk menemukan keseimbangan optimal antara kompleksitas model dan akurasi klasifikasi. Pemilihan *hyperparameter* dan nilai dari masing-masing *hyperparameter* ditunjukkan pada Tabel 4.2.

Tabel 4. 2 Penggunaan Hyperparameter dan Kombinasi Nilai

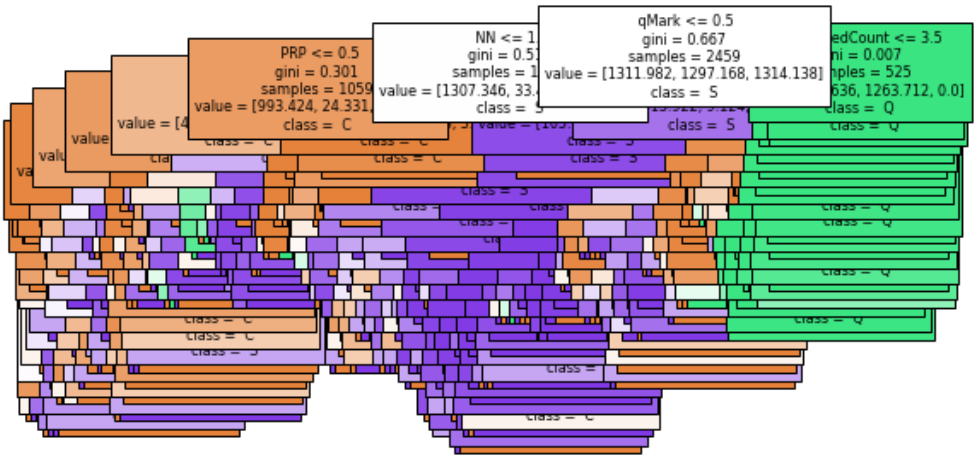
Jumlah pohon (*n_estimator*)	Kedalaman pohon (*max_depth*)	Jumlah minimum sampel untuk membagi node (*min_sample_split*)
100, 200, 500	5, 10, 20	2, 5, 10

Setelah parameter ditentukan, proses pembentukan model dilakukan menggunakan bahasa pemrograman *Python* dengan pustaka *scikit-learn*. Pustaka ini menyediakan modul *RandomForestClassifier* yang memudahkan implementasi algoritma. Dataset yang telah dibagi menjadi data latih dan data uji dimasukkan ke dalam fungsi pelatihan. Pada fungsi tersebut, parameter *n_estimators*, *max_depth*, dan *min_samples_split* diatur sesuai dengan kombinasi yang ingin diuji. Dengan satu baris perintah, model *Random Forest* dapat dibentuk sesuai dengan pengaturan parameter yang ditentukan. Proses ini diulang untuk setiap kombinasi parameter sehingga menghasilkan beberapa model. Penerapan parameter dalam proses pembentukan model ditunjukkan pada Gambar 4.6.

```
from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier(
                        n_estimators=500, #100, 200, 500
                        criterion='gini',
                        max_depth=20, #5, 10, 20
                        min_samples_split=10, #2, 5, 10
                        min_samples_leaf=1,
                        max_features='sqrt',
                        class_weight='balanced',
                        bootstrap=True,
                        random_state=100)
```

Gambar 4. 6 Penerapan Parameter dalam Proses Pembentukan Model

Proses *training* data dilakukan dengan cara memberikan data latih ke model *Random Forest* yang telah dibentuk. Pada tahap ini, setiap pohon keputusan dalam model belajar dari subset data yang dipilih secara acak dengan teknik *bootstrap*. Pohon juga memilih subset fitur secara acak untuk menentukan aturan klasifikasi. Dengan demikian, setiap pohon membentuk jalur keputusan yang unik berdasarkan variasi data dan fitur. Proses *training* ini menghasilkan kumpulan pohon dengan struktur aturan yang berbeda-beda. Kumpulan pohon inilah yang nantinya akan digunakan untuk memprediksi data uji. Contoh hasil *training* data berupa kumpulan pohon dalam proses pembentukan model ditunjukkan pada Gambar 4.7.



Gambar 4. 7 Contoh Hasil *Training* Data Berupa Kumpulan Pohon

Dalam penelitian ini, model klasifikasi yang dibentuk benar-benar memanfaatkan dataset percakapan yang telah dilabeli. Misalnya, pada data latih terdapat kalimat "*How are you?*" dengan label *Question*. Pohon keputusan yang dilatih akan belajar bahwa kalimat dengan kata tanya "*how*" berasosiasi dengan label *Question*. Demikian juga, kalimat seperti "*The meeting starts at 9 AM*" akan dipelajari sebagai *Statement* karena memiliki pola dominan kata benda dan angka.

Contoh lain, kalimat ringan seperti "*Hello, what's up?*" sering diasosiasikan dengan kategori *Chat*. Semua pola ini diolah bersama dalam pohon-pohon klasifikasi.

Proses pembentukan model juga menekankan pada integrasi informasi bahasa, baik dari segi struktur, fungsi, maupun penggunaannya dari *POS tagging*. Label seperti NN, NNP, NNPS, CD, WP, dan WRB menjadi fitur tambahan dalam proses klasifikasi. Misalnya, keberadaan label WP dalam sebuah kalimat sering kali menjadi indikator kuat untuk kategori *Question*. Sebaliknya, keberadaan NN atau NNP mendominasi kategori *Statement*. Sementara itu, kategori *Chat* sering kali memiliki variasi sederhana dengan kosakata informal. Dengan menggunakan informasi ini, pohon-pohon dalam *Random Forest* dapat membentuk aturan yang lebih kontekstual dalam membedakan kategori kalimat.

Setelah *training* selesai, setiap model diuji untuk memastikan bahwa proses pembentukan berhasil. Model yang terbentuk menyimpan aturan-aturan klasifikasi dari setiap pohon. Dengan *Python*, hasil model dapat diekspor atau divisualisasikan untuk melihat struktur pohon. Proses ini menunjukkan bagaimana model benar-benar terbentuk berdasarkan dataset percakapan. Hasil training inilah yang akan digunakan pada tahap berikutnya, yaitu pengujian terhadap data uji. Dengan demikian, pembentukan model klasifikasi selesai dilakukan sesuai dengan urutan proses yang telah dirancang.

4.2.3 Pembangunan Chatbot

Pembangunan chatbot berbasis Telegram dalam penelitian ini bertujuan untuk mengimplementasikan model klasifikasi *Random Forest* ke dalam sistem percakapan nyata. Chatbot yang diberi nama Gumara dirancang untuk mampu

memahami masukan pengguna berupa kalimat, kemudian mengklasifikasikannya ke dalam kategori tertentu. Dengan menggunakan *Telegram Bot API*, chatbot dapat diakses secara langsung melalui aplikasi Telegram pada perangkat pengguna. Proses pembangunan dimulai dari pembuatan bot baru menggunakan layanan *BotFather* untuk mendapatkan *token API*. Token ini digunakan sebagai autentikasi antara kode program dengan server Telegram. Inisialisasi Token API Telegram ditunjukkan pada gambar 4.8

```
@logger_config.logger
def __init__(self, TOKEN: str) -> None:
    super(TelegramBot, self).__init__()
    self.TOKEN = TOKEN
    self.URL = "https://api.telegram.org/bot6254423152:AAEiYXxaT9NQ_nl_JLfJSUpzD88zbggo3M8/"
    logging.debug("Telegram Bot ready")
```

Gambar 4. 8 Inisialisasi Token API Telegram

Integrasi model dilakukan menggunakan bahasa pemrograman *Python* karena kemampuannya yang fleksibel dalam mengelola *API* dan *machine learning*. Library *python-telegram-bot* digunakan untuk menangani komunikasi dengan server Telegram. Model *Random Forest* yang sudah dilatih pada tahap sebelumnya diimpor ke dalam skrip Python agar dapat digunakan langsung pada saat prediksi. *Pipeline preprocessing* yang sama dengan tahap pelatihan juga dimasukkan ke dalam program, sehingga input pengguna diproses secara konsisten. *Preprocessing* ini mencakup pembersihan teks, tokenisasi, *POS tagging*, *stemming*, dan penghapusan *stopwords*. Setelah diproses, input pengguna diubah menjadi representasi vektor *TF-IDF*. Vektor inilah yang kemudian dimasukkan ke model untuk menghasilkan prediksi kategori kalimat. Setelah semua tahapan pembentukan model klasifikasi selesai dibentuk, maka model klasifikasi dipanggil

dengan menyertakan kata kunci pola dari kalimat yang sudah ditentukan. Implementasi model klasifikasi Random Forest pada chatbot ditunjukkan pada gambar 4.9.

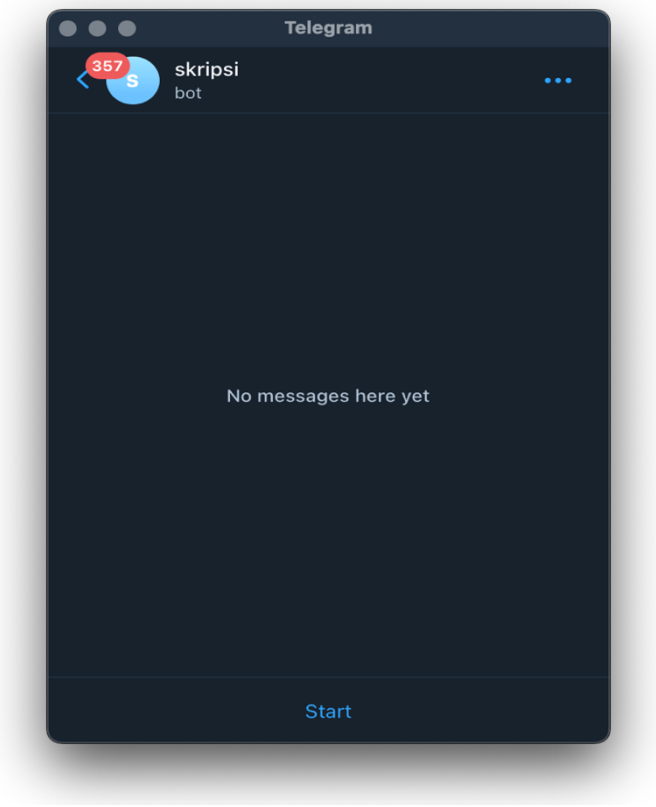
```python
@logger_config.logger
def setup():
    utilities.setup_nltk()
    logging.debug("NLTK setup completed")

    model_file = "model.joblib"
    retrain = False
    if Path(model_file).exists():
        last_modified_time = Path(model_file).stat().st_mtime
        time_now = time.time()
        diff_in_days = (time_now - last_modified_time) // (86400)
        RETRAIN_AFTER_DAYS = 7
        if diff_in_days < RETRAIN_AFTER_DAYS:
            logging.debug("Loading pre-trained model")
            clf = load(model_file)
        else:
            retrain = True
    else:
        retrain = True
    if retrain:
        logging.debug("Training model")
        clf = utilities.classify_model()
        # clf = utilities.classify_model_adv(model="rf")
        dump(clf, model_file)
    logging.debug("Classification model ready")

    databaseconnect.setup_database()
    logging.debug("Database setup completed, database connected")
    learn_response = LearnResponse.MESSAGE.name
    return clf, learn_response
```

Gambar 4. 9 Implementasi Model Klasifikasi *Random Forest* pada Chatbot

Chatbot Gumara dirancang untuk memberikan respon berdasarkan hasil klasifikasi model. Jika input dikategorikan sebagai *Question*, *chatbot* akan membalas dengan jawaban atau informasi yang sesuai. Jika input berupa *Statement*, maka *chatbot* akan memberikan konfirmasi atau penegasan ulang. Sedangkan pada kategori *Chat*, sistem memberikan balasan ringan atau sapaan untuk menjaga percakapan tetap natural. Dengan pembagian respon ini, chatbot diharapkan mampu

memberikan interaksi yang lebih relevan sesuai kebutuhan pengguna. Pengaturan balasan juga dibuat dalam bentuk kamus respon agar mudah dikembangkan.

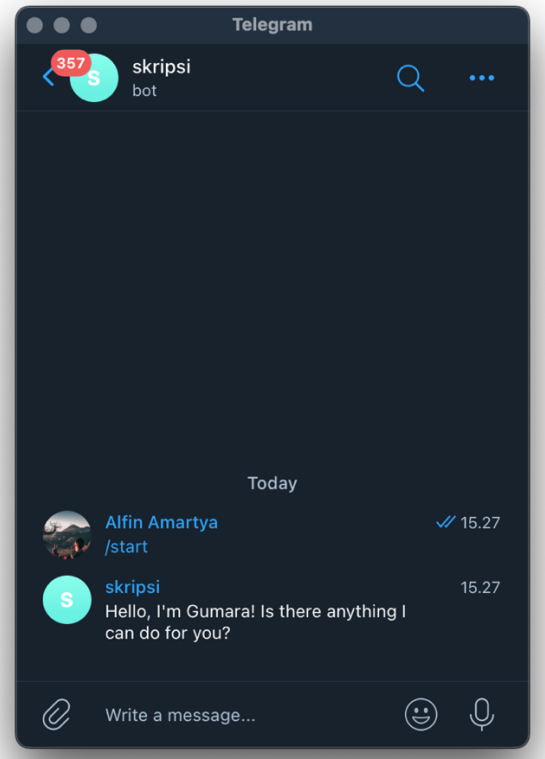
Tampilan antarmuka chatbot di Telegram sangat sederhana karena memanfaatkan desain asli aplikasi Telegram. Pengguna hanya perlu mengetikkan pesan ke dalam kolom chat dan menunggu respon dari bot. Respon yang diberikan muncul dalam bentuk balasan teks langsung pada jendela percakapan. Tampilan utama pada sistem yang dibangun ditunjukan oleh Gambar 4.10.



Gambar 4. 10 Tampilan Utama pada Telegram

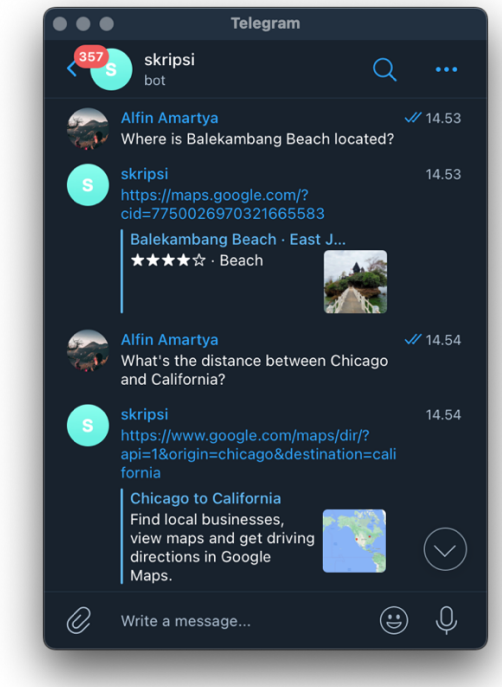
Pada tampilan utama chatbot virtual route guide, terdapat button */start* yang berfungsi untuk memulai percakapan atau interaksi dengan bot dengan mengkliknya atau dengan mengetikkan perintah */start* pada jendela obrolan

telegram. Setelah mengirimkan perintah */start,* chatbot akan merespon dengan mengirimkan kalimat balasan berupa greeting sebagai awal percakapan antara pengguna dengan bot. Tampilan berupa pesan balasan awal bot pada sistem yang dibangun ditunjukan oleh Gambar 4.11.

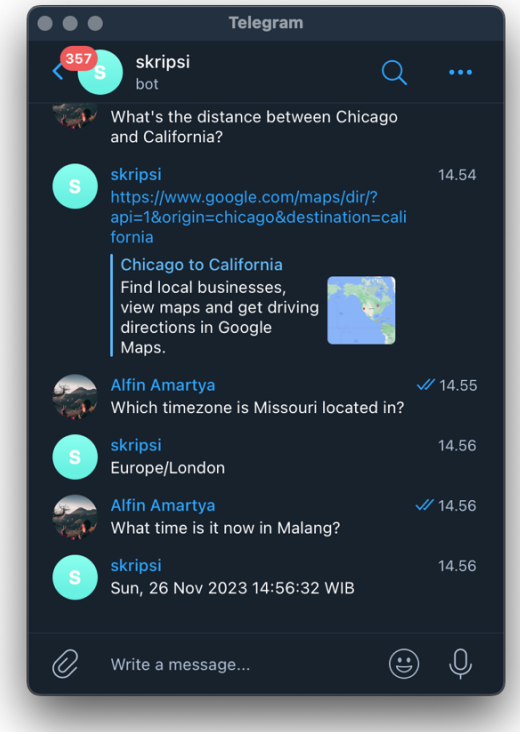


Gambar 4. 11 Tampilan Pesan Balasan Awal pada Bot

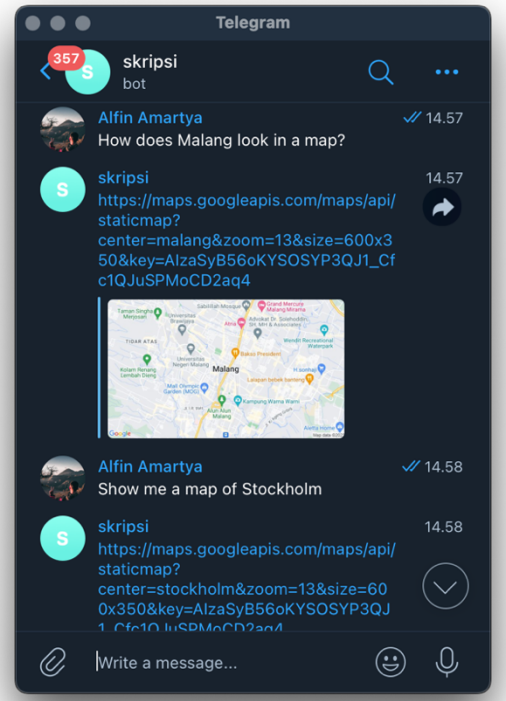
Selanjutnya pengguna bisa berinteraksi dengan bot, dengan mengirimkan pesan percakapan tantang menanyakan lokasi yang ingin dituju, jarak antara lokasi pengguna dengan lokasi yang dituju, waktu yang ada pada lokasi pengguna sekarang, dan gambaran peta dari lokasi yang diinginkan. Percakapan antara pengguna dengan bot ditunjukkan pada Gambar 4.12, Gambar 4.13, dan Gambar 4.14.



Gambar 4. 12 Percakapan Tentang Lokasi dan Jarak Tujuan



Gambar 4. 13 Percakapan Tentang Waktu di Lokasi Tujuan



Gambar 4. 14 Percakapan Tentang Peta Lokasi Tujuan

4.3 Hasil Uji Coba

Pada bab ini membahas secara menyeluruh mengenai performa sistem klasifikasi yang telah dibangun menggunakan algoritma *Random Forest Classifier*. Pada bagian ini dijelaskan bagaimana model yang telah dibentuk diuji menggunakan data uji yang terpisah dari data latih untuk menilai kemampuan generalisasi. Proses uji coba dilakukan dengan mengatur berbagai kombinasi parameter, termasuk jumlah pohon, kedalaman pohon, dan jumlah minimum sampel split, untuk menemukan konfigurasi yang paling optimal. Evaluasi performa model dilakukan menggunakan metrik akurasi serta didukung dengan analisis *confusion matrix* untuk melihat distribusi prediksi tiap kelas. Selain itu, disajikan

pula contoh hasil klasifikasi pada kalimat tertentu sebagai bukti konkret cara kerja model dalam mengenali pola bahasa.

4.3.1 Evaluasi Model Klasifikasi

Evaluasi model merupakan tahap penting dalam penelitian ini karena menjadi dasar untuk menilai performa algoritma *Random Forest Classifier*. Proses evaluasi dilakukan dengan menggunakan dataset uji sebesar 30% dari total data yang telah dipisahkan sebelumnya. Data uji ini dipilih secara *stratified sampling* untuk menjaga proporsi setiap kelas, sehingga distribusi *Statement*, *Question*, dan *Chat* tetap seimbang. Langkah ini penting agar evaluasi tidak bias pada kelas tertentu saja. Dengan demikian, hasil pengujian benar-benar mencerminkan kemampuan generalisasi model. Evaluasi model dilakukan dengan cara menghitung akurasi serta menganalisis hasil prediksi yang dibandingkan dengan label asli.

Hasil pengujian kombinasi parameter disajikan dalam bentuk tabel akurasi untuk memperlihatkan performa model pada setiap konfigurasi. Tabel ini memuat sembilan kombinasi parameter inti yang dipilih dari total kemungkinan kombinasi. Penyajian tabel memudahkan pembaca untuk membandingkan pengaruh masing-masing parameter terhadap akurasi. Dengan table ini, dapat diketahui bahwa kedalaman pohon memiliki peran besar dalam meningkatkan performa model. Selain itu, jumlah pohon yang lebih banyak cenderung memberikan stabilitas hasil meskipun perbedaan akurasi tidak terlalu signifikan. Hasil akurasi dengan kombinasi parameter ditunjukkan pada Tabel 4.3.

Tabel 4. 3 Hasil Akurasi Kombinasi Hyperparameter

n_estimator	max_depth	min_sample_split	Akurasi
500	5	2	0,5686
	10	5	0,6911
	20	10	0,8028
200	5	2	0,5418
	10	5	0,6930
	20	10	0,8028
100	5	2	0,5309
	10	5	0,6732
	20	10	0,8021

Berdasarkan tabel di atas, terlihat bahwa akurasi terbaik dicapai pada konfigurasi dengan *n_estimators* sebesar 200 atau 500, *max_depth* sebesar 20, dan min_samples_split sebesar 10. Kombinasi parameter ini menghasilkan akurasi 0.8028 atau 80,28% yang merupakan hasil tertinggi dari seluruh percobaan. Perbedaan kecil pada nilai *n_estimators* menunjukkan bahwa jumlah pohon yang terlalu besar tidak selalu meningkatkan akurasi secara signifikan. Namun, kedalaman pohon yang lebih besar terbukti membantu model menangkap pola yang lebih kompleks dalam kalimat. Nilai *min_samples_split* sebesar 10 juga menjaga agar pembagian *node* tidak terlalu mudah dilakukan. Kombinasi ini menghasilkan model yang seimbang antara detail aturan dan kemampuan generalisasi.

Analisis hasil menunjukkan bahwa parameter *max_depth* memberikan pengaruh dominan terhadap performa model. Pada konfigurasi dengan *max_depth* sebesar 5, akurasi berada di kisaran 53–56%, yang menunjukkan model gagal menangkap pola kalimat yang kompleks. Ketika *max_depth* ditingkatkan menjadi 10, akurasi naik menjadi 67–69%, menunjukkan perbaikan yang signifikan. Namun, hasil terbaik baru diperoleh ketika *max_depth* mencapai 20 dengan akurasi lebih dari 80%. Hal ini menegaskan bahwa dataset percakapan ini membutuhkan kedalaman pohon yang cukup besar untuk mengenali pola pembentukan kalimat dengan baik. Dengan demikian, kedalaman pohon optimal sangat penting dalam klasifikasi teks.

Selain kedalaman pohon, parameter *min_samples_split* juga terbukti berperan dalam menjaga stabilitas hasil. Nilai *min_samples_split* yang rendah, seperti 2, cenderung membuat pohon membagi *node* terlalu cepat, sehingga menghasilkan akurasi yang rendah. Ketika nilai ini ditingkatkan menjadi 5, akurasi mengalami peningkatan, meskipun belum optimal. Nilai terbaik diperoleh ketika *min_samples_split* ditetapkan pada 10, di mana model menjadi lebih selektif dalam membagi *node*. Dengan demikian, parameter *min_samples_split* berkontribusi dalam menjaga keseimbangan model.

Untuk memberikan gambaran detail mengenai cara kerja model, disajikan tabel hasil klasifikasi pada kalimat contoh dari data uji. Tabel ini memperlihatkan kalimat asli, label yang asli, dan hasil prediksi dari model dengan konfigurasi terbaik. Contoh ini juga menunjukkan bahwa model mampu menangani variasi

kalimat yang berbeda. Hasil klasifikasi dari contoh data uji ditunjukkan pada Tabel 4.4.

Tabel 4. 4 Hasil Klasifikasi dari Contoh Data Uji

Kalimat	Label Asli	Prediksi Model
How do you feel today?	Question	Question
The meeting starts at 9 AM	Statement	Statement
Hello, good to see you	Chat	Chat
Where is the nearest station?	Question	Question
I can't remember personal details	Chat	Chat

.

Dari tabel contoh hasil klasifikasi di atas, terlihat bahwa model dapat mengenali pola kalimat dengan baik. Kalimat yang diawali kata tanya seperti "*How*" dan "*Where*" berhasil diklasifikasikan sebagai *Question*. Kalimat dengan pola deskriptif seperti "*The meeting starts at 9 AM*" tepat diklasifikasikan sebagai *Statement*. Sementara itu, kalimat ringan seperti "*Hello, good to see you*" diklasifikasikan sebagai *Chat*.

4.3.2 Analisis Hasil Uji Coba

Analisis hasil uji coba pada penelitian ini menggunakan *confusion matrix* sebagai alat utama untuk menilai performa model. *Confusion matrix* memberikan gambaran detail mengenai jumlah prediksi yang benar maupun salah pada setiap kelas. Dengan menggunakan data uji sebesar 30% dari total dataset, diperoleh distribusi hasil prediksi untuk kategori *Chat* (C), *Question* (Q), dan *Statement* (S). Model yang digunakan pada tahap ini adalah *Random Forest Classifier* dengan

parameter terbaik, yaitu *n_estimators* = 200, *max_depth* = 20, dan *min_samples_split* = 10. Hasil *confusion matrix* ditunjukkan pada tabel 4.5.

Tabel 4. 5 Hasil Klasifikasi dari Contoh Data Uji

	Pred_C	**Pred_Q**	**Pred_S**
Actual_C	763	13	3
Actual_Q	200	143	3
Actual_S	90	0	352

Dari tabel *confusion matrix* di atas, dapat dilihat bahwa kelas *Chat* memiliki jumlah prediksi benar terbanyak dengan 763 data teridentifikasi sesuai label aslinya. Namun, terdapat 13 data *Chat* yang salah diprediksi sebagai *Question* dan 3 data lainnya sebagai *Statement*. Kelas *Question* relatif lebih sulit diprediksi dengan baik, di mana hanya 143 data yang benar, sementara 200 data salah diklasifikasikan sebagai *Chat*. Untuk kelas Statement, model menunjukkan kinerja cukup baik dengan 352 data terprediksi benar, meskipun masih ada 90 data yang salah diklasifikasikan sebagai Chat. Distribusi ini memperlihatkan variasi kesulitan pada tiap kelas. Analisis lebih dalam diperlukan untuk memahami penyebab kesalahan tersebut.

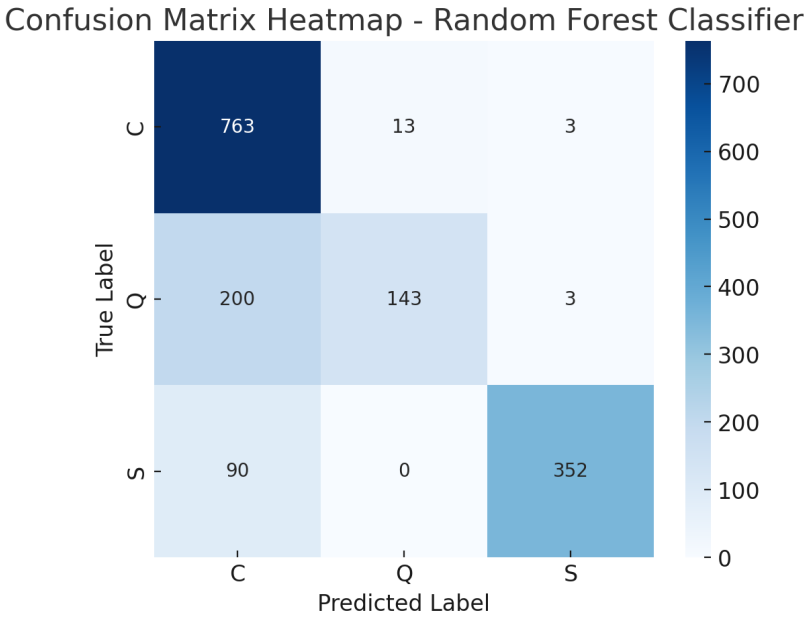
Nilai akurasi dapat dihitung dengan menggunakan rumus berikut:

$$Akurasi = \frac{Jumlah\ Prediksi\ Benar}{Total\ Prediksi} \times 100\%$$

Dengan menggunakan nilai dari *confusion matrix*, jumlah prediksi benar adalah 763 + 143 + 352 = 1258. Jumlah total data uji adalah 1567. Sehingga nilai akurasi yang diperoleh adalah:

$$Akurasi = \frac{1258}{1567} \times 100\% = 80{,}28\%$$

Hasil ini sesuai dengan hasil evaluasi parameter terbaik yang telah diperoleh sebelumnya. Angka ini menunjukkan bahwa model memiliki performa cukup baik dalam mengenali pola kalimat percakapan. Hasil *confusion matrix* menggunakan data uji coba ditunjukkan pada Gambar 4.15.



Gambar 4. 15 Hasil *Confusion Matrix* Menggunakan Data Uji Coba

Distribusi hasil prediksi menunjukkan bahwa kategori *Chat* adalah yang paling mudah dikenali oleh model. Hal ini dapat dipahami karena kalimat *Chat* cenderung memiliki struktur sederhana, kosakata informal, serta pola berulang yang mudah ditangkap oleh algoritma. Sebaliknya, kategori *Question* merupakan yang paling sulit diprediksi dengan akurasi rendah. Banyak kalimat *Question* salah

diklasifikasikan sebagai *Chat* karena kemiripan struktur pada kalimat tanya informal. Sementara itu, kategori *Statement* berada di posisi tengah dengan performa cukup baik, meskipun masih ada kesalahan prediksi. Hasil ini mengindikasikan bahwa tingkat kesulitan prediksi dipengaruhi oleh variasi struktur kalimat. Faktor yang berhubungan dengan bahasa dari *POS tagging* berperan besar dalam perbedaan hasil prediksi antar kelas. Kategori *Question* biasanya ditandai oleh keberadaan kata tanya seperti "*what*", "*where*", atau "*how*" yang diberi label WP atau WRB. Namun, dalam beberapa kasus, pertanyaan dengan bentuk tidak baku cenderung lebih mirip kalimat *Chat* sehingga membingungkan model. Kategori *Statement* lebih banyak mengandung kata benda (NN, NNP) atau angka (CD) yang jelas mengindikasikan informasi faktual. Sedangkan kategori *Chat* sering kali terdiri dari kosakata sederhana dengan label POS yang lebih bervariasi.

Hasil analisis ini juga memberikan implikasi praktis terhadap penggunaan model. Untuk meningkatkan akurasi pada kelas *Question*, perlu dipertimbangkan penambahan fitur lain selain *POS tagging*, seperti analisis dependensi sintaksis atau penggunaan *embedding* berbasis konteks. Meskipun performa pada kelas Chat sudah baik, variasi kosakata yang terus berubah memerlukan model yang adaptif. Sementara itu, kelas *Statement* sudah menunjukkan performa stabil karena pola kalimatnya lebih teratur. Analisis ini membuka peluang pengembangan model ke arah yang lebih baik. Secara keseluruhan, analisis hasil uji coba dengan *confusion matrix* memperlihatkan bahwa *Random Forest Classifier* mampu mencapai akurasi sebesar 80,28% pada dataset percakapan. Kategori *Chat* menjadi kelas yang paling mudah diprediksi, sedangkan *Question* merupakan kelas yang paling menantang.

Kesalahan prediksi sebagian besar muncul akibat overlap pola antar kelas. Pemilihan parameter optimal terbukti meningkatkan stabilitas akurasi dan kualitas hasil.

4.4 Pembahasan

Pembahasan hasil uji coba pada penelitian ini bertujuan untuk menekankan hubungan antara performa model *Random Forest Classifier* dengan tujuan penelitian yang telah dirumuskan. Berdasarkan hasil uji coba, model terbaik mencapai akurasi 80,28% menggunakan parameter *n_estimators* = 200, *max_depth* = 20, dan *min_samples_split* = 10. Nilai akurasi ini menunjukkan bahwa model sudah cukup baik untuk mengenali pola bahasa dalam percakapan. Hal ini selaras dengan hipotesis awal penelitian bahwa *Random Forest* dapat menangani variasi data teks dengan baik.

Hasil uji coba juga menggambarkan bagaimana model yang dibangun mampu bekerja secara konsisten dalam mengenali pola bahasa. *Random Forest* yang dibangun dari ratusan pohon keputusan mampu menyerap variasi dalam data percakapan dengan tingkat kestabilan yang cukup tinggi. Penggunaan parameter optimal membuat model mampu menjaga keseimbangan antara bias dan variansi, sehingga prediksi lebih akurat. Model tidak hanya menghafal pola pada data latih, tetapi juga mampu mengeneralisasi pada data uji yang belum pernah dilihat. Hal ini terlihat dari distribusi hasil klasifikasi yang relatif seimbang pada tiap kelas. Dengan demikian, model ini memenuhi kebutuhan penelitian dalam menghasilkan sistem klasifikasi teks yang dapat digunakan untuk aplikasi nyata.

Analisis hasil evaluasi model memperlihatkan bahwa variasi parameter berpengaruh signifikan terhadap performa model. Kombinasi *max_depth* = 5 menghasilkan akurasi rendah, sekitar 53–56%, karena kedalaman yang dangkal gagal menangkap kompleksitas bahasa. Kombinasi *max_depth* = 10 meningkatkan akurasi hingga kisaran 67–69%, namun hasil terbaik baru diperoleh pada kedalaman 20. Jumlah pohon (*n_estimators*) juga memengaruhi stabilitas prediksi, di mana 200 atau 500 pohon memberikan hasil yang konsisten dengan selisih akurasi kecil. Nilai *min_samples_split* = 10 terbukti mampun untuk membantu model untuk lebih selektif dalam membagi *node*, sehingga aturan yang terbentuk lebih kontekstual. Dengan pengaturan parameter yang optimal, model dapat mencapai performa yang seimbang antara akurasi dan efisiensi.

Interpretasi *confusion matrix* memberikan gambaran lebih detail mengenai distribusi hasil prediksi. Tabel berikut menyajikan hasil confusion matrix pada data uji dengan parameter terbaik.

Tabel 4. 6 Hasil Klasifikasi Data Uji Menggunakan Parameter Terbaik

	Pred_C	**Pred_Q**	**Pred_S**
Actual_C	763	13	3
Actual_Q	200	143	3
Actual_S	90	0	352

Berdasarkan tabel 4.6., terlihat bahwa kelas *Chat* memiliki prediksi benar terbanyak, yaitu 763 data. Namun, terdapat kesalahan di mana 13 data *Chat* diprediksi sebagai *Question* dan 3 sebagai *Statement*. Kelas *Question* menjadi yang

paling sulit karena dari total 346 data, hanya 143 yang tepat, sementara 200 salah sebagai *Chat*. Sementara itu, kelas *Statement* menunjukkan performa baik dengan 352 prediksi benar dari 442 data. Distribusi ini mengungkapkan tantangan utama dalam membedakan Question dari Chat.

Meskipun demikian, penngaruh *POS tagging* pada proses klasifikasi terbukti memperkaya representasi fitur model. Dengan *POS tagging*, setiap token dalam kalimat tidak hanya dikenali secara makna dasar, tetapi juga fungsinya dalam struktur kalimat. Misalnya, keberadaan kata dengan label WP (*wh-pronoun*) atau WRB (*wh-adverb*) menjadi penanda kuat untuk kalimat *Question*. Label NN (*noun*) dan NNP (*proper noun*) sering mendominasi kalimat Statement yang bersifat informatif. Sedangkan kalimat *Chat* lebih variatif dengan campuran kata kerja sederhana (VB), kata ganti (PRP), dan ekspresi informal. Dengan tambahan informasi ini, pohon-pohon dalam *Random Forest* dapat membentuk aturan klasifikasi yang lebih tajam. Hal ini terbukti dari akurasi yang lebih tinggi dibandingkan jika *POS tagging* tidak digunakan.

Analisis fitur terhadap ciri-ciri khas yang menyusun sebuah kalimat menunjukkan bahwa kelas *Statement* didominasi oleh NN, NNP, dan CD (*cardinal number*) yang berhubungan dengan informasi faktual. Kelas *Question* sangat dipengaruhi oleh kata-kata tanya dengan label WP (*who, what, which*) dan WRB (*where, when, how*). Sementara itu, kelas *Chat* banyak mengandung PRP (*personal pronoun*) seperti “I”, “you”, serta ekspresi singkat tanpa struktur yang formal. Perbedaan dominasi fitur ini menjelaskan mengapa kelas *Statement* lebih mudah diprediksi dibanding *Question*. Model cenderung kesulitan saat *Question* berbentuk

informal tanpa kata tanya eksplisit, sehingga sering salah diprediksi sebagai *Chat*. Dengan demikian, variasi fitur linguistik menjadi salah satu faktor utama kesalahan klasifikasi. Hubungan antara pola linguistik dengan kesalahan klasifikasi dapat dilihat dari distribusi error. Kalimat tanya informal seperti "*you okay?*" atau "*coming now?*" sering salah diklasifikasikan sebagai *Chat*. Hal ini terjadi karena tidak adanya kata tanya eksplisit yang dapat dikenali model sebagai indikator *Question*. Sebaliknya, kalimat *Chat* seperti "*I don't know*" terkadang diklasifikasikan sebagai *Statement* karena strukturnya mirip pernyataan. Kesalahan ini menunjukkan adanya overlap pola linguistik antar kelas. Oleh karena itu, meskipun *POS tagging* membantu, tetap ada keterbatasan dalam membedakan kalimat yang bersifat ambigu.

Analisis kesalahan (*error analysis*) juga memperlihatkan bahwa kelas *Question* memiliki error tertinggi, sementara *Chat* memiliki error terendah. Hal ini konsisten dengan hasil *confusion matrix* yang menunjukkan bahwa 200 data *Question* salah diklasifikasikan sebagai *Chat*. Penyebab utamanya adalah variasi kosakata dan struktur kalimat tanya yang lebih luas dibanding kelas lain. Beberapa pertanyaan singkat tidak memiliki ciri linguistik kuat yang bisa dipelajari model. Sebaliknya, *Chat* yang lebih informal cenderung lebih konsisten dalam pola linguistiknya sehingga mudah dikenali. Hasil ini menunjukkan bahwa tingkat kesulitan prediksi tidak sama antar kelas.

Pengaruh variasi parameter terhadap kestabilan hasil juga terlihat jelas dari analisis evaluasi. Jumlah pohon yang terlalu sedikit membuat model kurang stabil, sementara jumlah pohon yang cukup banyak memberikan hasil lebih konsisten.

Namun, peningkatan jumlah pohon dari 200 ke 500 tidak lagi memberikan peningkatan signifikan, menunjukkan adanya titik jenuh. Kedalaman pohon yang besar sangat penting untuk menangkap pola kompleks dalam data percakapan. Nilai *min_samples_split* yang lebih tinggi membantu model menghindari percabangan yang tidak perlu. Dengan pengaturan parameter yang maksimal, model akan bisa mencapai kombinasi terbaik antara akurasi dan efisiensi.

Penilaian konsistensi model menunjukkan bahwa hasil prediksi stabil ketika parameter berada pada konfigurasi tertentu. Dengan kombinasi *n_estimators* = 200 dan *max_depth* = 20, akurasi tercatat konstan di 80%. Hal ini berbeda dengan konfigurasi *max_depth* = 5 yang menghasilkan variasi hasil lebih besar. Konsistensi ini menunjukkan bahwa model lebih andal ketika diberi ruang cukup untuk membentuk aturan mendalam. Dengan demikian, konfigurasi optimal dapat dijadikan acuan untuk penelitian lanjutan.

Secara keseluruhan, model yang dibangun memiliki kekuatan dalam mengenali pola bahasa sehari-hari, terutama pada kalimat *Chat* dan *Statement*. Akurasi tinggi pada kedua kelas ini membuktikan bahwa model mampu menangkap ciri linguistik sederhana dan faktual. Tantangan utama masih berada pada kelas Question yang membutuhkan pendekatan tambahan seperti *word embedding* berbasis konteks untuk meningkatkan performa. Meskipun demikian, hasil penelitian ini tetap relevan untuk implementasi praktis pada sistem berbasis percakapan seperti chatbot. Dengan adanya kontribusi *POS tagging*, model ini lebih kontekstual dalam mengenali fungsi linguistik kalimat. Oleh karena itu, penelitian ini berhasil menunjukkan efektivitas *Random Forest* dalam klasifikasi percakapan.

4.5 Integrasi Keilmuan Dalam Islam

Keakuratan dan kebenaran dalam memberikan informasi merupakan prinsip utama dalam Islam. Sebagaimana firman Allah SWT dalam Al-Qur'an Surat Al-Maidah ayat 8:

يٰٓاَيُّهَا الَّذِيْنَ اٰمَنُوْا كُوْنُوْا قَوّٰمِيْنَ لِلّٰهِ شُهَدَاۤءَ بِالْقِسْطِۖ وَلَا يَجْرِمَنَّكُمْ شَنَاٰنُ قَوْمٍ عَلٰٓى اَلَّا تَعْدِلُوْاۗ اِعْدِلُوْاۗ هُوَ اَقْرَبُ لِلتَّقْوٰىۖ وَاتَّقُوا اللّٰهَۗ اِنَّ اللّٰهَ خَبِيْرٌۢ بِمَا تَعْمَلُوْنَ

"*Hai orang-orang yang beriman, hendaklah kamu jadi orang-orang yang selalu menegakkan keadilan, menjadi saksi karena Allah meskipun terhadap dirimu sendiri, atau ibu bapak dan kaum kerabatmu. Jika ia kaya atau miskin, maka Allah lebih tahu kemaslahatannya. Maka janganlah kamu mengikuti hawa nafsu karena ingin menyimpang dari kebenaran. Dan jika kamu memutar balikkan (kata-kata atau keterangan) atau enggan memberinya kesaksiannya, maka sesungguhnya Allah Maha Mengetahui apa yang kamu kerjakan*" (Q.S. Al-Maidah : 8).

Ayat ini memerintahkan kepada orang mukmin agar melaksanakan amal dan pekerjaan mereka dengan cermat, jujur dan ikhlas karena Allah SWT, baik pekerjaan yang bertalian dengan urusan agama maupun pekerjaan yang bertalian dengan urusan kehidupan duniawi. Karena hanya dengan demikianlah mereka bisa sukses dan memperoleh hasil atau balasan yang mereka harapkan. Dalam persaksian, mereka harus adil menerangkan apa yang sebenarnya, tanpa memandang siapa orangnya, sekalipun akan menguntungkan lawan dan merugikan sahabat dan kerabat. Ayat ini juga menunjukkan bahwa dalam memberikan informasi, haruslah berpegang teguh pada kebenaran dan keadilan. Oleh karena itu, dalam penggunaan *chatbot*, integrasi keilmuan Islam perlu diterapkan untuk menjaga keakuratan informasi yang diberikan *chatbot* kepada pengguna.

Dalam muamalah, informasi adalah instrumen amal yang bermanfaat jika dijalankan sesuai etika syariah. Memberikan informasi termasuk praktik sosial yang

mencerminkan hubungan dengan Allah SWT dan sesama manusia. Disebut hubungan dengan Allah SWT (Ma'Allah), ketika informasi disampaikan dengan niat tulus dan ikhlas semata-mata untuk meraih ridho Allah SWT serta untuk menegakkan kebenaran. Memberikan informasi yang benar tanpa pamrih antar sesama manusia harus berdasar pada syariat Islam yang meliputi kejujuran, adil dan amanah. Oleh karena itu, memberikan informasi merupakan bagian dari muamalah yang mencerminkan implementasi ketulusan ibadah (muamalah Ma'Allah) serta keadilan dalam interaksi (muamalah Ma'annas). Sehingga menghasilkan komunikasi yang bermanfaat, amanah, dan berguna bagi setiap masyarakat.

Dalam tafsir Jalalain menjelaskan bahwa dalam ayat tersebut, Allah SWT memerintahkan kita yang beriman untuk selalu berdiri menegakkan kebenaran dan menjadi saksi dengan adil. Kita tidak boleh terdorong oleh kebencian terhadap suatu kaum sehingga kita menjadi tidak adil terhadap mereka. Keadilan adalah lebih dekat kepada ketakwaan. Oleh karena itu, kita harus berlaku adil baik terhadap lawan maupun kawan, karena Allah SWT Maha Mengetahui apa yang kita kerjakan. Hal ini penting untuk diingat agar kita selalu bertakwa kepada Allah SWT dan menerima pembalasan dari-Nya (As-Suyuthi & Al-Mahally, 1505).

Allah SWT juga berfirman di dalam Al-Qur'an pada Surat Al-An'am ayat 152-153:

وَلَا تَقرَبُوا مَالَ اليَتِيمِ اِلَّا بِالَّتِى هِىَ اَحسَنُ حَتّٰى يَبلُغَ اَشُدَّهٗ ۚ وَاَوفُوا الكَيلَ وَالمِيزَانَ
بِالقِسطِ ۚ لَا نُكَلِّفُ نَفسًا اِلَّا وُسعَهَا ۚ وَاِذَا قُلتُم فَاعدِلُوا وَلَو كَانَ ذَا قُربٰى ۚ وَبِعَهدِ اللّٰهِ اَوفُوا ۭ
لِكُم وَصّٰىكُم بِهٖ لَعَلَّكُم تَذَكَّرُونَ ۙ ﴿١٥٢﴾ وَاَنَّ هٰذَا صِرَاطِى مُستَقِيمًا فَاتَّبِعُوهُ ۚ وَلَا تَتَّبِعُوا
السُّبُلَ فَتَفَرَّقَ بِكُم عَن سَبِيلِهٖ ۭ لِكُم وَصّٰىكُم بِهٖ لَعَلَّكُم تَتَّقُونَ ﴿١٥٣﴾

"*Dan janganlah kamu mendekati harta anak yatim, kecuali dengan cara yang lebih bermanfaat, sampai dia mencapai (usia) dewasa. Dan sempurnakanlah takaran dan timbangan dengan adil. Kami tidak membebani seseorang melainkan menurut kesanggupannya. Apabila kamu berbicara, bicaralah sejujurnya, sekalipun dia kerabat (mu) dan penuhilah janji Allah. Demikianlah Dia memerintahkan kepadamu agar kamu ingat. Dan sungguh, inilah jalan-Ku yang lurus. Maka ikutilah! Jangan kamu ikuti jalan-jalan (yang lain) yang akan mencerai-beraikan kamu dari jalan-Nya. Demikianlah Dia memerintahkan kepadamu agar kamu bertakwa*" (Q.S. Al-An'am : 152-153).

Ayat ini menegaskan bahwa dalam Islam kebenaran dan keadilan harus dipegang teguh dalam segala aspek kehidupan, termasuk dalam memberikan informasi. Oleh karena itu, dalam penggunaan *chatbot* integrasi keilmuan Islam tentang keakuratan informasi perlu diterapkan untuk menjaga kualitas dan kejujuran dalam memberikan informasi kepada pengguna.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian dan implementasi, dapat disimpulkan bahwa penelitian ini berhasil merancang dan membangun sebuah sistem *Chatbot Virtual Route Guide* bernama "Gumara" yang memanfaatkan algoritma *Random Forest Classifier* dalam proses klasifikasinya. Sistem ini dikembangkan dengan tujuan untuk memberikan panduan percakapan yang mampu mengklasifikasikan input pengguna ke dalam kategori *Statement*, *Question*, dan *Chat*. Proses klasifikasi didukung oleh preprocessing data teks, meliputi pembersihan, tokenisasi, *stemming*, *stopword removal*, serta *POS tagging*. Hasil uji coba menunjukkan bahwa model *Random Forest* dapat mengenali pola bahasa pengguna dengan cukup akurat. Hal ini membuktikan bahwa algoritma ini efektif diterapkan dalam pemrosesan bahasa alami untuk sistem chatbot. Dengan demikian, penelitian ini berhasil memenuhi tujuan awalnya dalam membangun chatbot berbasis NLP.

Dari hasil evaluasi, diperoleh akurasi terbaik sebesar 80,28% pada konfigurasi parameter *n_estimators* = 200, *max_depth* = 20, dan *min_samples_split* = 10. Nilai ini menunjukkan bahwa pemilihan parameter yang tepat berpengaruh signifikan terhadap performa sistem. *Confusion matrix* dengan menghitung niyang dihasilkan memperlihatkan bahwa kategori *Chat* memiliki tingkat klasifikasi tertinggi, sedangkan kategori *Question* menjadi yang paling sulit dibedakan. Performa sistem ini selaras dengan karakteristik linguistik dataset, di mana kalimat

percakapan informal cenderung lebih mudah dikenali dibanding pertanyaan formal atau semi-formal. Dengan akurasi ini, chatbot Gumara mampu memberikan respon yang cukup tepat pada sebagian besar interaksi. Hasil ini memperkuat keyakinan bahwa metode *Random Forest* dapat diandalkan untuk kebutuhan sistem klasifikasi percakapan.

Penerapan *POS tagging* memberikan kontribusi penting dalam meningkatkan kualitas klasifikasi. Dengan adanya label kata seperti NN, NNP, WP, dan WRB, sistem mampu mengenali pola kalimat lebih baik dibanding hanya berbasis kata mentah. Kelas *Statement* lebih mudah dikenali karena dominasi kata benda dan angka, sedangkan kelas *Question* sangat dipengaruhi oleh keberadaan kata tanya. Namun, penelitian ini juga menemukan adanya keterbatasan dalam mengenali pertanyaan singkat yang sering salah diklasifikasikan sebagai *Chat*. Hal ini menunjukkan bahwa meskipun *POS tagging* membantu, masih diperlukan pengembangan fitur tambahan seperti *word embeddings* berbasis konteks untuk meningkatkan akurasi. Dengan begitu, pengolahan linguistik menjadi faktor kunci dalam pengembangan chatbot berbasis NLP.

Secara keseluruhan, penelitian ini berhasil menunjukkan bahwa *Chatbot Virtual Route Guide* "Gumara" dapat berfungsi dengan baik menggunakan metode *Random Forest Classifier*. Sistem ini mampu memproses input teks, mengklasifikasikan percakapan, dan memberikan output sesuai kategori dengan tingkat akurasi yang memadai. Kekuatan utama sistem terletak pada kemampuannya mengenali pola kalimat sehari-hari, terutama dalam kategori *Chat* dan *Statement*. Keterbatasan utama terletak pada kesalahan prediksi pada kategori

Question, yang masih memerlukan peningkatan. Dengan demikian, penelitian ini dapat dijadikan landasan untuk pengembangan chatbot lebih lanjut yang lebih cerdas dan adaptif.

5.2 Saran

Dalam perancangan dan pembangunan *Chatbot Virtual Route Guide* pada penelitian ini, terdapat beberapa saran yang dapat diterapkan oleh peneliti selanjutnya agar mencapai hasil yang lebih maksimal, yaitu:

a. Perlu dilakukan perluasan dataset dengan jumlah data yang lebih banyak dan lebih bervariasi, terutama pada kategori *Question*, agar model lebih mampu mengenali pola pertanyaan informal.
b. Disarankan untuk menambahkan metode *word embedding* seperti *Word2Vec* atau *BERT* yang berguna untuk memperkaya representasi teks dan meningkatkan akurasi klasifikasi.
c. Perlu pengujian dengan algoritma lain seperti *SVM*, *Gradient Boosting*, atau *Deep Learning* sebagai pembanding performa dengan *Random Forest*.
d. Sistem dapat dikembangkan dengan menambahkan modul *intent recognition* agar chatbot tidak hanya mengenali setiap kategori, tetapi juga bisa memahami maksud dari pengguna.
e. Perlu ditambahkan fitur *context awareness* agar chatbot mampu mempertahankan konteks percakapan dalam interaksi yang lebih panjang.

f. Disarankan adanya pembaruan dan pelatihan ulang model secara berkala agar *chatbot* dapat menyesuaikan diri dengan perkembangan bahasa percakapan yang dinamis.
g. Disarankan agar sistem dikembangkan lebih lanjut dengan mendukung fitur multibahasa, khususnya Bahasa Indonesia, untuk memperluas aksesibilitas dan meningkatkan pengalaman pengguna lokal.
h. Disarankan agar sistem diperluas ke platform lain seperti *WhatsApp*, *Facebook Messenger*, atau website interaktif dengan mempertimbangkan penggunaan *framework* yang mendukung *multi-platform deployment* untuk menjangkau lebih banyak pengguna.

DAFTAR PUSTAKA

Abilowo, K., Santoni, M. M., & Muliawati, A. (2020). Perancangan Chatbot Sebagai Pembelajaran Dasar Bahasa Jawa Menggunakan Artificial Intelligence Markup Language. *Informatik: Jurnal Ilmu Komputer*, *16*(3), 139–147.

Anton, M. (2015). *Laporan Kinerja Tahunan 2014*. Http://Malangkota.Go.Id.

As-Suyuthi, J., & Al-Mahally, J. M. I. A. (1505). *Terjemah - Tafsir Jalalain 30 Juz*.

Badan Pusat Statistik Kota Batu. (2021). *Jumlah Pengunjung Objek Wisata dan Wisata Oleh-oleh Menurut Tempat Wisata di Kota Batu*.

Baiti, Z. N. (2013). *Aplikasi chatbot "MI3" untuk informasi jurusan teknik informatika berbasis sistem pakar menggunakan metode forward chaining*. Universitas Islam Negeri Maulana Malik Ibrahim.

Bichri, H., Chergui, A., & Hain, M. (2024). Investigating the Impact of Train / Test Split Ratio on the Performance of Pre-Trained Models with Custom Datasets. In *IJACSA) International Journal of Advanced Computer Science and Applications* (Vol. 15, Issue 2). www.ijacsa.thesai.org

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (2017). *Classification and regression trees* (1st ed.). Routledge.

Criminisi, A., Shotton, J., & Konukoglu, E. (2012). Decision Forests: A Unified Framework for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning. *Foundations and Trends® in Computer Graphics and Vision*, *7*(2–3), 81–227.

Dana Vrajitoru. (2004). Evolutionary Sentence Combination for Chatterbots. *Computer and Information Sciences Indiana University South Bend*.

Devella, S., Yohannes, Y., & Rahmawati, F. N. (2020). Implementasi Random Forest Untuk Klasifikasi Motif Songket Palembang Berdasarkan SIFT. *JATISI (Jurnal Teknik Informatika Dan Sistem Informasi)*, *7*(2), 310–320.

Hamburger, & Elise. (2014, February 25). *Why Telegram has become the hottest messaging app in the world*. Http://Www.Theverge.Com/2014/2/25/5445864/Telegram-Messenger-Hottest-App-in-the-World.

Ismayanti. (2010). Pengantar pariwisata. *PT Gramedia Widisarana*.

Izquierdo-Verdiguier, E., & Zurita-Milla, R. (2020). An evaluation of Guided Regularized Random Forest for classification and regression tasks in remote sensing. *International Journal of Applied Earth Observation and Geoinformation*, *88*, 102051. https://doi.org/10.1016/J.JAG.2020.102051

Kemenparekraf/Baparekraf. (2022, August 18). *Ada 6 Juta Wisatawan Berkunjung ke Kota Malang, Jumlahnya Lampaui Target 2022*.

Liu, S., Xu, L., Li, D., Li, Q., Jiang, Y., Tai, H., & Zeng, L. (2013). Prediction of dissolved oxygen content in river crab culture based on least squares support vector regression optimized by improved particle swarm optimization. *Computers and Electronics in Agriculture*, *95*, 82–91.

Maryono. (2016, December 6). *Malang Raya*. Cakmaryono.Com/Malang-Raya/.

Maskur. (2016). *Perancangan Chatbot Pusat Informasi Mahasiswa Menggunakan Aiml Sebagai Virtual Assistant Berbasis Web*. Universitas Muhammadiyah Malang.

Muhamad, D. A. (2020). *Deteksi Distributed Denial Of Service Menggunakan Random Forest*. Universitas Islam Negeri Maulana Malik Ibrahim.

Mustafa, M. S., Ramadhan, M. R., & Thenata, A. P. (2018). Implementasi Data Mining untuk Evaluasi Kinerja Akademik Mahasiswa Menggunakan Algoritma Naive Bayes Classifier. *Creative Information Technology Journal*, *4*(2), 151–162.

Nugroho. (2020). BEBERAPA MASALAH DALAM PENGEMBANGAN SEKTOR PARIWISATA DI INDONESIA. *Pariwisata*, *7*(2), 124–131.

Patel, R. K., & Giri, V. K. (2016). Feature selection and classification of mechanical fault of an induction motor using random forest classifier. *Perspectives in Science*, *8*, 334–337. https://doi.org/10.1016/j.pisc.2016.04.068

Pemerintah Kabupaten Malang. (2018). *Kabupaten Malang Satu Data : BAB 23 Pariwisata*.

Pemerintah Kota Malang. (2021). *Malang Satu Data : Pariwisata*.

Pinto, R. L. (2014). *Secure Instant Messaging* [Master Thesis]. Frankfurt University.

Putra, F. H. (2017). PENGEMBANGAN SISTEM PENDUKUNG KEPUTUSAN PEMILIHAN OBJEK WISATA DI MALANG RAYA DENGAN MENGGUNAKAN METODE ANALYTICAL HIERARCHY PROCESS (AHP). *JATI (Jurnal Mahasiswa Teknik Informatika*, *1*(1), 462–469.

Stefan Wager. (n.d.). *Asymptotic Theory for Random Forests | Enhanced Reader*. https://doi.org/https://doi.org/10.48550/arXiv.1405.0352

Sun, Z., Wang, G., Li, P., Wang, H., Zhang, M., & Liang, X. (2024). An improved random forest based on the classification accuracy and correlation measurement of decision trees. *Expert Systems with Applications*, *237*, 121549. https://doi.org/10.1016/J.ESWA.2023.121549

Sutton, C. D. (2005). Classification and Regression Trees, Bagging, and Boosting. *Handbook of Statistics*, *24*, 303–329. https://doi.org/10.1016/S0169-7161(04)24011-1

Syukron, A., & Subekti, A. (2018). Penerapan Metode Random Over-Under Sampling dan Random Forest Untuk Klasifikasi Penilaian Kredit. *Jurnal Informatika*, *5*(2), 175–185.

Vrigazova, B. (2021). The Proportion for Splitting Data into Training and Test Set for the Bootstrap in Classification Problems. *Business Systems Research*, *12*(1), 228–242. https://doi.org/10.2478/bsrj-2021-0015

Weizenbaum, J. (1966). *ELIZA-A computer program for the study of natural language communication between man and machine* (8th ed., Vol. 10). Communicationsof the ACM.

Widmaier, F. (2015). *Robot Arm Tracking with Random Decision Forests*. Universität Tübingen Tübingen.

Wirajaya, Y. (2013). ANALISIS KEPUASAN WISATAWAN MANCANEGARA TERHADAP KUALITAS PELAYANAN PARIWISATA. *Jurnal Manajemen Dan Akuntansi*, *2*(3), 95–109.

Yuda Wenika. (2014). Panduan wisata Jepang, Tokyo, Osaka, dan Kyoto. *Bhuana Ilmu Populer*.