TESIS

ANALISIS SENTIMEN PUBLIK TERHADAP ELEKTABILITAS GANJAR PRANOWO DI TAHUN POLITIK 2024 DI TWITTER DENGAN ALGORITMA KNN DAN NAÏVE BAYES

Disusun oleh:

Nama : **Dede Sandi**
NIM : **21.55.1072**
Konsentrasi : **Business Intelligence**

PROGRAM STUDI S2 TEKNIK INFORMATIKA

PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA

YOGYAKARTA

2023

TESIS

ANALISIS SENTIMEN PUBLIK TERHADAP ELEKTABILITAS GANJAR PRANOWO DI TAHUN POLITIK 2024 DI TWITTER DENGAN ALGORITMA KNN DAN NAÏVE BAYES

PUBLIC SENTIMENT ANALYSIS OF GANJAR PRANOWO'S ELECTABILITY IN THE 2024 POLITICAL YEAR ON TWITTER USING THE KNN AND NAÏVE BAYES ALGORITHM

Diajukan untuk memenuhi salah satu syarat memperoleh derajat Magister

Disusun oleh:

Nama : **Dede Sandi**
NIM : **21.55.1072**
Konsentrasi : **Business Intelligence**

PROGRAM STUDI S2 TEKNIK INFORMATIKA

PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA

YOGYAKARTA

2023

HALAMAN PENGESAHAN

ANALISIS SENTIMEN PUBLIK TERHADAP ELEKTABILITAS GANJAR PRANOWO DI TAHUN POLITIK 2024 DI TWITTER DENGAN ALGORITMA KNN DAN NAÏVE BAYES

PUBLIC SENTIMENT ANALYSIS OF GANJAR PRANOWO'S ELECTABILITY IN THE 2024 POLITICAL YEAR ON TWITTER USING KNN AND NAÏVE BAYES ALGORITHM

Dipersiapkan dan Disusun oleh

Dede Sandi

21.55.1072

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada hari Senin, 6 November 2023

Tesis ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Magister Komputer

Yogyakarta, 6 November 2023
Rektor

Prof. Dr. M. Suyanto, M.M.
NIK. 190302001

HALAMAN PERSETUJUAN

ANALISIS SENTIMEN PUBLIK TERHADAP ELEKTABILITAS GANJAR PRANOWO DI TAHUN POLITIK 2024 DI TWITTER DENGAN ALGORITMA KNN DAN NAÏVE BAYES

PUBLIC SENTIMENT ANALYSIS OF GANJAR PRANOWO'S ELECTABILITY IN THE 2024 POLITICAL YEAR ON TWITTER USING KNN AND NAÏVE BAYES ALGORITHM

Dipersiapkan dan Disusun oleh

Dede Sandi

21.55.1072

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada hari Senin, 6 November 2023

Pembimbing Utama

Prof. Dr. Ema Utami, S.Si., M.Kom.
NIK. 190302035

Anggota Tim Penguji

Prof. Dr. Kusrini, M.Kom.
NIK. 190302106

Alva Hendi M, S.T., M.Eng., Ph.D.
NIK. 1903024930

Pembimbing Pendamping

Kusnawi, S.Kom., M.Eng.
NIK. 190302112

Prof. Dr. Ema Utami, S.Si., M.Kom.
NIK. 190302035

Tesis ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Magister Komputer

Yogyakarta, 6 November 2023
Direktur Program Pascasarjana

Prof. Dr. Kusrini, M.Kom.
NIK. 190302106

HALAMAN PERNYATAAN KEASLIAN TESIS

Yang bertandatangan di bawah ini,

Nama mahasiswa : Dede Sandi
NIM : 21.55.1072
Konsentrasi : Business Intelligence

Menyatakan bahwa Tesis dengan judul berikut:
Analisis Sentimen Publik Terhadap Elektabilitas Ganjar Pranowo Di Tahun Politik 2024 Di Twitter Dengan Algoritma KNN Dan Naïve Bayes

Dosen Pembimbing Utama : Prof. Dr. Ema Utami, S.Si., M.Kom.
Dosen Pembimbing Pendamping : Kusnawi, S.Kom., M.Eng.

1. Karya tulis ini adalah benar-benar ASLI dan BELUM PERNAH diajukan untuk mendapatkan gelar akademik, baik di Universitas AMIKOM Yogyakarta maupun di Perguruan Tinggi lainnya.
2. Karya tulis ini merupakan gagasan, rumusan dan penelitian SAYA sendiri, tanpa bantuan pihak lain kecuali arahan dari Tim Dosen Pembimbing.
3. Dalam karya tulis ini tidak terdapat karya atau pendapat orang lain, kecuali secara tertulis dengan jelas dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan disebutkan dalam Daftar Pustaka pada karya tulis ini.
4. tanggung jawab SAYA, bukan tanggung jawab Universitas AMIKOM Yogyakarta.
5. Pernyataan ini SAYA buat dengan sesungguhnya, apabila di kemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka SAYA bersedia menerima SANKSI AKADEMIK dengan pencabutan gelar yang sudah diperoleh, serta sanksi lainnya sesuai dengan norma yang berlaku di Perguruan Tinggi.

Yogyakarta, 6 November 2023
Yang Menyatakan,

Dede Sandi

HALAMAN PERSEMBAHAN

بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ

Alhamdulillah, atas izin Tuhan Alam Semesta Allah SWT, penelitian tesis ini telah selesai yang juga merupakan suatu kebahagiaan serta kebanggaan karena bisa menempuh pada fase ini. Tesis ini saya persembahkan untuk Ibu, Ayah, dan Adikku tersayang serta Sang Masa Depan Istri dan Anak-anak semua keturunan saya yang akan menjadi *legacy* semua ilmu pengetahuan di masa yang akan datang.

Tak lupa pula, tesis ini saya persembahkan kepada Almamater S1 tercinta FKom Universitas Kuningan khususnya Prodi Teknik Informatika yang telah memberikan kepercayaan untuk menerima Beasiswa S2 Alumni Berprestasi.

Terakhir, penelitian ini juga saya persembahkan untuk Almamater tercinta, tempat belajar, bertumbuh, dan berkembang Universitas Amikom Yogyakarta serta para Pembaca yang istimewa semoga penelitian ini dapat bermanfaat serta dapat dikembangkan lebih baik lagi.

HALAMAN MOTTO

KATA PENGANTAR

بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ

Alhamdulillah, senantiasa kita panjatkan puja dan puji serta syukur kepada Sang Tuhan Alam Semesta Allah SWT yang telah memberikan segala karunia dan nikmat-Nya yang tak pernah habis hingga saat ini, sehingga tesis yang berjudul "**Analisis Sentimen Publik Terhadap Elektabilitas Ganjar Pranowo Di Tahun Politik 2024 Di Twitter Dengan Algoritma KNN Dan Naïve Bayes**" telah selesai dengan tepat waktu tentu berkat nikmat dan rida yang telah diberikan oleh-Nya.

Melalui halaman ini, izinkan Penulis untuk menyampaikan ungkapan terima kasih yang tak terhingga kepada:

1. Kedua Orang Tua Ibu dan Ayah yang senantiasa memberikan doa dan dukungan terbaiknya yang tak lekang oleh waktu.
2. Adikku tersayang Desi Puspa Rahmawati, S.Pd., MCE., yang juga atas doa dan dukungan terbaiknya.
3. Bapak Prof. Dr. Mohammad Suyanto, M.M., selaku Rektor Universitas Amikom Yogyakarta.
4. Ibu Prof. Dr. Kusrini, M.Kom., selaku Direktur Pascasarjana Universitas Amikom Yogyakarta sekaligus Penguji Utama pada ujian tesis yang telah memberikan arahan, masukan, dan perbaikan agar penelitian ini menjadi lebih baik.

5. Ibu Prof. Dr. Ema Utami, S.Si., M.Kom., selaku Dosen Pembimbing Utama sekaligus Penguji Ketiga pada ujian tesis yang telah memberikan bimbingan terbaiknya dan ilmu serta arahan beliau dalam penelitian ini.
6. Bapak Kusnawi, S.Kom., M.Eng., selaku Dosen Pembimbing Pendamping yang telah memberikan masukan-masukan terkait Penelitian ini.
7. Bapak Alva Hendi Muhammad, S.T., M.Eng., Ph.D., selaku Penguji Kedua pada ujian tesis yang telah banyak memberikan pandangan serta masukan-masukan yang membangun.
8. Segenap Dosen dan Staf Magister Teknik Informatika Universitas Amikom Yogyakarta yang telah memberikan ilmu, wawasan, bantuan, pengalaman hingga petuah-petuah selama masa kuliah.
9. Rekan-rekan seperjuangan Magister Teknik Informatika PJJ dari sabang sampai merauke khususnya kawan-kawan konsentrasi *Business Intelligence* yang senantiasa saling bahu-membahu dalam menyelesaikan studi perkuliahan.
10. Terakhir, tentu kepada diri Saya sendiri yang telah tangguh berjuang sampai pada titik ini dengan berbagai kausalitas realitas untuk menjadi nyata.

Yogyakarta, 6 November 2023

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

INTISARI

Tahun politik untuk di tahun 2024 kini telah semakin menjamu bagi segenap masyarakat Indonesia untuk melangsungkan pesta demokrasi oleh segenap masyarakat terkait elektabilitas masing-masing calon presiden. Salah satu di antaranya ialah Ganjar Pranowo yang merupakan tokoh politik Gubernur Jawa Tengah yang akhir-akhir ini di pertengahan tahun 2023 telah diusung oleh salah satu partai politik untuknya maju ke kancah kursi kepala negara sebagai calon presiden untuk pemilihan di tahun 2024 mendatang.

Dengan adanya berbagai polemik pendapat dari berbagai lapisan masyarakat, ini merupakan momentum tepat untuk melakukan analisis sebagai bentuk kebulatan polarisasi yang demikian tersuguhkan dari berbagai opini masyarakat sebagai gambaran secara umum dan garis besar dalam bersentimen berupa informasi atas kesimpulan opini publik. Pada tahap eksperimen, penulis melakukan pembagian data dengan persentase 80% data training dan 20% data testing. Pemodelan yang digunakan ialah K-Nearest Neighbor (KNN) dan Naïve Bayes untuk mengklasifikasikan data teks sekaligus melakukan perbandingan atas keduanya.

Dalam proses implementasinya, penulis menggunakan python sebagai bahasa pemrograman dalam membangun model tersebut. Dari hasil pengujian atas perbandingan dari kedua pemodelan tersebut, pemodelan K-Nearest Neighbor yang paling baik akurasinya dengan nilai accuracy sebesar 99% dari K-Nearest Neighbor dengan nilai accuracy sebesar 96%. Persentase sentiment dengan perbandingan 96,6% sentimen positif dan 3,4% sentimen negatif menyimpulkan bahwa sebagian besar masyarakat masih mendominasi sentimen positif.

Kata Kunci: Ganjar Pranowo; Analisis Sentimen; *Twitter*; *K-Nearest Neighbor*; *Naïve Bayes*

ABSTRACT

The political year 2024 has now become more welcoming for all Indonesian people to hold a democratic party for all people regarding the electability of each presidential candidate. One of them is Ganjar Pranowo, who is a political figure, Governor of Central Java, who recently in mid-2023 was promoted by a political party for him to advance to the position of head of state as a presidential candidate for the 2024 elections.

With the existence of various polemics of opinion from various layers of society, this is the right moment to carry out an analysis as a form of polarization unanimity which is presented from various public opinions as a general description and an outline in sentiment in the form of information on the conclusions of public opinion. At the experimental stage, the authors divided the data with a percentage of 80% training data and 20% testing data. The modeling used is K-Nearest Neighbor (KNN) and Naïve Bayes to classify text data as well as make comparisons of the two.

In the implementation process, the author uses python as a programming language in building the model. From the test results on the comparison of the two models, the K-Nearest Neighbor model has the best accuracy with an accuracy value of 99% of the K-Nearest Neighbor with an accuracy value of 96%. The percentage of sentiment with a comparison of 96.6% positive sentiment and 3.4% negative sentiment concluded that most people still dominate positive sentiment.

***Keywords**: Ganjar Pranowo; Sentiment Analysis; Twitter, K-Nearest Neighbor; Naïve Bayes*

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Elektabilitas dalam dunia politik merupakan sebuah sebuah perbincangan hangat yang menjadi fokus sentral masyarakat secara indisental dalam pengaruh kepada individu atau bahkan pihak tertentu. Setiap kabar atau pun informasi teraktual yang mudah sekali didapat dan di berbagai sumber informasi digital membuat masyarakat gemar memberikan tanggapan yang seringkali diartikan sebagai *feedback* untuk tokoh-tokoh atau pihak tertentu. Ganjar Pranowo merupakan tokoh politik Gubernur Jawa Tengah yang menjabat dua periode sejak 23 Agustus 2023. Di penghujung masa kepemimpinannya, Ganjar Pranowo telah diusung menjadi salah satu Calon Presiden di tahun 2024 mendatang. Rekam jejak Ganjar Pranowo yang beberapa tahun ini yang telah menjadi salah satu pusat perhatian masyarakat, sehingga membuatnya semakin dikenal dan didekati masyarakat dari berbagai kalangan. Sebelumnya, Ganjar Pranowo menjabat sebagai Anggota Dewan Perwakilan Rakyat (DPR) Republik Indonesia dengan masa jabatan selama sembilan tahun dari 1 Oktober 2004 sampai 23 Agustus 2013 melalui daerah pemilihan Jawa Tengah VII. Selain pasangan capres-cawapres Prabowo-Gibran dan Anies-Amin, Ganjar-Mahfud semakin ramai diperbincangkan oleh masyarakat untuk menuju tahun politik 2024 mendatang. Berbagai polemik antara pendapat dan opini dari berbagai lapisan masyarakat yang terjadi sekarang ini merupakan momentum tepat untuk melakukan analisis sebagai bentuk kebulatan

polarisasi yang demikian tersuguhkannya berbagai opini publik sebagai gambaran secara umum dan juga sebagai garis besar dalam bersentimen agar publik dapat memperoleh informasi atas kesimpulan opini masyarakat, khususnya untuk capres Ganjar Pranowo yang disandingkan dengan wacapres Mahfud MD.

Twitter merupakan media sosial yang seringkali menjadi pusat *trending* mengenai isu di dunia skala nasioal maupun internasional yang dijadikan warganet sebagai media untuk menyuarakan opini terkait sentimen terhadap apa pun yang terkini diperbincangkan di jejaring sosial yang begitu kompleks. Pengguna *twitter* di Indonesia memiliki 19,5 juta pengguna total 500 juta pengguna global.

Sentiment Analysis adalah sentimen dari teks subjektif tersebut menganalisis, memproses, meringkas, dan proses inferensial. *Sentiment analysis* saat ini dibagi menjadi penggunaan klasifikasi pembelajaran mesin dan metode klasifikasi berdasarkan aturan, metode pembelajaran mesin menggunakan kata kata emosi sebagai klasifikasi fitur, dan kamus emosi dapat digunakan untuk mewujudkan pemilihan karakteristik sentimen dengan cepat dan efisien (Isnain dkk., 2021).

K-Nearst Neighbors (KNN) adalah merupakan metode yang termasuk ke dalam algoritma *supervised* yang mana data baru akan diklasifikan dengan didasarkan mayoritas kelas tertentu dengan tujuan pengklasifikasian suatu objek baru atas mayoritas pada kategori tersebut.

Naïve Bayes merupakan metode algoritma untuk memprediksi peluang di masa yang akan datang berdasarkan pengalaman di masa sebelumnya dengan menghitung probabilitas bahwa kelas keputusan adalah benar mengasumsikan bahwa atribut objek yang independen.

Berdasarkan latar belakang yang telah dikemukakan, maka penelitian ini menggunakan *K-Nearest Neighbors* (KNN) dan *Naïve Bayes* sebagai komparasi untuk mengetahui perbandingan hasil akurasi klasifikasi opini sentimen yang lebih akurat.

1.2. Rumusan Masalah

Berdasarkan latar belakang masalah yang telah dijelaskan di atas, maka rumusan masalah dalam penelitian ini adalah :

a. Bagaimana peningkatan komparasi akurasi algoritma *K-Nearest Neighbors* (KNN) dan *Naïve Bayes* dalam memproses data untuk analisis sentimen?
b. Bagaimana sentimen publik terhadap Elektabilitas Ganjar Pranowo Di Tahun Politik 2024?

1.3. Batasan Masalah

Pembatasan suatu masalah dilakukan untuk menghindari penyimpangan maupun pelebaran inti masalah agar penelitian lebih terarah dalam pembahasan sehingga tujuan penelitian dapat tercapai. Beberapa batasan masalah dalam penelitian ini adalah :

a. Pengambilan data hanya pada media sosial *twitter*.
b. *Tweet* yang diambil merupakan *tweet* yang menggunakan Bahasa Indonesia.
c. Dataset yang diambil ialah *tweet* di *twitter* yang tersedia dimulai tanggal 1 Januari 2023 sampai 12 Mei 2023.

d. Menggunakan metode *K-Nearest Neighbors* (KNN) dan *Naïve Bayes* dalam memproses data teks opini publik dari media sosial *twitter*.

e. Sentimen terdiri dari tiga kelas yaitu positif, negatif, dan netral.

1.4. Tujuan Penelitian

Adapun tujuan penulis dalam melakukan penelitian ini adalah :

a. Mengklasifikasikan sentimen opini publik di *Twitter* menggunakan *K-Nearest Neighbors* (KNN) dan *Naïve Bayes*.

b. Melakukan analisis terhadap hasil perhitungan metode *K-Nearest Neighbors* (KNN) dan *Naïve Bayes*.

c. Mengetahui hasil komparasi akurasi yang optimal menggunakan *K-Nearest Neighbors* (KNN) dan *Naïve Bayes*.

1.5. Manfaat Penelitian

Manfaat dari penelitian ini adalah :

a. Mempermudah dalam proses pengklasifikasian opini di *Twitter* berbahasa Indonesia.

b. Mengetahui kinerja dan tingkat komparasi akurasi klasifikasi *text* menggunakan algoritma *K-Nearest Neighbors* (KNN) dan algoritma *Naïve Bayes*.

c. Memberikan informasi mengenai reaksi sentimen publik kepada pembaca terhadap Elektabilitas Ganjar Pranowo Di Tahun Politik 2024.

BAB II

TINJAUAN PUSTAKA

2.1. Tinjauan Pustaka

Beberapa peneliti sebelumnya telah melakukan penelitian menggunakan algoritma *K-Nearest Neighbors* (KNN) maupun *Naïve Bayes*. Penelitian yang dilakukan oleh Achmad Bayhaqy, dkk, pada tahun 2018, dengan judul "*Sentiment Analysis about E-Commerce from Tweets Using Decision Tree, K-Nearest Neighbor, and Naïve Bayes*" (Bayhaqy dkk., t.t.). Penelitian ini bertujuan untuk membandingkan *result* untuk mengetahui *classifier* yang memberikan hasil terbaik dalam hal rasio *recall* dan akurasi menggunakan *data mining*. Hasil penelitian menunjukkan bahwa akurasi dari *Decision Tree, K-NN,* dan *Naïve Bayes* sebesar 80%, 78%, dan 77%. Hasil Presisi *Decision Tree, K-NN,* dan *Naïve Bayes* sebesar 79,96%, 85,67%, dan 88,50%. Itu hasil juga menunjukkan bahwa *recall* dari *Decision Tree, K-NN,* dan *Naïve Bayes* adalah 84%, 70%, dan 64%. Jadi bisa disimpulkan bahwa pengklasifikasi *Naïve Bayes* adalah pengklasifikasi terbaik untuk digunakan dengan kumpulan data media sosial karena menyediakan lebih akurat dan prediksi yang tepat.

Penelitian yang dilakukan oleh Akhmad Deviyanto dan Muhammad Didik Rohmad Wahyudi pada tahun 2018, dengan judul "*Penerapan Analisis Sentimen pada Pengguna Twitter Menggunakan Metode K-Nearest Neighbor*" (Deviyanto dkk., 2018). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet* yang mengandung kata 'AHY' dari mulai tanggal 1 Januari 2017 sampai

dengan 31 Januari 2017. Hasil penelitian menunjukkan bahwa akurasi terbesar adalah 67,2% dengan nilai k=5. Sedangkan nilai presisi tertinggi sebesar 56,94% saat k=5 dan *recall* terbesar 78,24 % ketika k=15.

Penelitian yang dilakukan oleh J. A. Septian, T. M. Fachrudin, dan A. Nugroho pada tahun 2019, dengan judul "*Analisis Sentimen Pengguna Twitter Terhadap Polemik Persepakbolaan Indonesia Menggunakan Pembobotan TF-IDF Dan K-Nearest Neighbor*" (Septian dkk., 2019). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet* dengan *query* '@PSII' dari mulai tanggal 21 Februari 2019 sampai dengan 4 Mei 2019. Hasil penelitian menunjukkan bahwa dari 2000 data *tweet* terkait polemik persepakbolaan Indonesia yang diambil dari akun twitter PSSI dengan proporsi 790 *tweet* positif dan 1210 *tweet* negatif, dilakukan percobaan untuk mencari model KNN dengan akurasi terbaik menggunakan range nilai k=1 hingga k=30 yang adalah bilangan ganjil, didapatkan akurasi optimal pada k=23 dengan akurasi sebesar 79,99% dan *error rate* sebesar 20,01%.

Penelitian yang dilakukan oleh Siti Saidah Salim dan Joanna Mayary pada tahun 2020, dengan judul "*Analisis Sentimen Pengguna Twitter Terhadap Dompet Elektronik Dengan Metode Lexicon Based Dan K-Nearest Neighbor*" (Salim & Mayary, 2020). Penelitian ini bertujuan untuk menganalisis sentimen di twitter dari data tweet dengan query 'OVO', 'Gopay', dan 'LinkAja'. Hasil penelitian menunjukkan bahwa pengujian menggunakan confusion matrix memperoleh hasil akurasi dari masing-masing data menggunakan metode *K-Nearest Neighbor* untuk *tweet* OVO cenderung positif ditunjukkan oleh nilai akurasi data 86,91% dan

kecenderungan negatif *tweet* OVO memiliki nilai akurasi data 13,09%, *tweet* Gopay cenderung positif dengan akurasi data 94,05% dan kecenderungan negatif *tweet* Gopay memiliki nilai akurasi data 5,95%, serta *tweet* LinkAja cenderung positif dengan akurasi data 76,31% dan kecenderungan negatif *tweet* LinkAja memiliki nilai akurasi data 23,69%.

Penelitian yang dilakukan oleh Dianati Duei Putri, Gigih Forda Nama, Wahyu Eko Sulistiono pada tahun 2022, dengan judul "*Analisis Sentimen Kinerja Dewan Perwakilan Rakyat (DPR) Pada Twitter Menggunakan Metode Naïve Bayes Classifier*" (Duei Putri dkk., 2022). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet* dengan *query* 'DPR'. Hasil penelitian menunjukkan bahwa *accuracy score* sebesar 0.8 atau 80% hal ini berarti sistem mampu memprediksi 80% secara akurat dari total data testing sebesar 20%. Berdasarkan hasil analisis sistem mendapatkan klasifikasi *tweet* dari *twitter* mengenai DPR sebanyak 95 positif, 693 netral dan 758 negatif dari data hasil *crawling* sebanyak 1546.

Penelitian yang dilakukan oleh Dedi Darwis, Nery Siskawati, dan Zaenal Abidin pada tahun 2022, dengan judul "*Penerapan Algoritma Naïve Bayes untuk Analisis Sentimen Review Data Twitter BMKG Nasional*" (Darwis dkk., t.t.). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet* dengan *query* 'BMKG'. Hasil penelitian menunjukkan bahwa proses ekstrakasi data dari Twitter BMKG Nasional menggunakan Bahasa pemrograman *Python 3.74* dengan tahapan *Preprocessing* yang meliputi *casefolding, filtering, tokenisasi,*

slang replacement dan *stopword removal*. Tingkat akurasi berdasarkan pengujian yang dilakukan adalah 68,97%.

Penelitian yang dilakukan oleh Fajar Ratnawati pada tahun 2018, dengan judul "*Implementasi Algoritma Naive Bayes Terhadap Analisis Sentimen Opini Film Pada Twitter*" (Fajar dkk., t.t.). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet* dengan *hashtag* '#judul film' dari mulai tanggal 1 Januari 2015 sampai dengan 31 Desember 2015. Hasil penelitian menunjukkan bahwa semakin banyak *data training* yang digunakan maka akan memengaruhi kinerja dari sistem. Hasil akurasi akan semakin tinggi dan itu menandakan sistem berhasil melakukan klasifikasi dengan baik. Akurasi tertinggi didapat pada *fold* kedua yaitu 90%, *precision* 92%, *recall* 90% dan *f-measure* 90%.

Penelitian yang dilakukan oleh Taofik Krisdiyanto dan Erry maricha Oki Nurharyanto pada tahun 2021, dengan judul "*Analisis Sentimen Opini Masyarakat Indonesia Terhadap Kebijakan PPKM pada Media Sosial Twitter Menggunakan Naïve Bayes Classifier*" (Krisdiyanto dkk., 2021). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet* dengan *hashtag* 'PPKM'dengan metode *crawling* secara otomatis menggunakan *software* RStudio. Hasil penelitian menunjukkan bahwa analisis sentimen yang sudah dilakukan menggunakan *software* RStudio terhadap data kata PPKM pada *twitter* yang berjumlah 1000 data ulasan, diperoleh hasil yaitu berupa sentimen positif sebanyak 99% dan sentimen negatif sebanyak 1% atau terlihat dari hasil klasifikasi emosi memiliki mayoritas "unknown" atau tidak diketahui yang sebagian besar kata tersebut memiliki polaritas positif.

Penelitian yang dilakukan oleh Dery Anjas Ramadhan dan Erwin Budi Setiawan pada tahun 2019, dengan judul "*Analisis Sentimen Program Acara Di SCTV Pada Twitter Menggunakan Metode Naïve Bayes Dan Support Vector Machine*" (Anjas Ramadhan & Budi Setiawan SSi, t.t.). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet* yang mengandung kata "ftv, sinetron, entertainmentsctv, dan liputan6sctv" dari mulai awal Mei 2019 sampai dengan akhir Juni 2019. Hasil penelitian menunjukkan bahwa akurasi terbaik antara metode *Naïve Bayes* dan *Support Vector Machine* adalah metode *Support Vector Machine* dengan Seluruh Program Acara didapatkan hasil akurasi 88,57% berikut perkategori yang diperoleh, Berita didapatkan hasil akurasi 79,81%, Entertaiment didapatkan hasil akurasi 89,80%, Sinetron didapatkan hasil akurasi 73,68% dan FTV didapatkan hasil akurasi 87,74%.

Penelitian yang dilakukan oleh Sri Rahayu, Yumarlin MZ, Jemmy Edwin Bororing, dan Rahmat Hidayat pada tahun 2022, dengan judul "*Implementasi Metode K-Nearest Neighbor (K-NN) untuk Analisis Sentimen Kepuasan Pengguna Aplikasi Teknologi Finansial FLIP*" (Rahayu dkk., 2022). Penelitian ini bertujuan untuk menganalisis sentimen data ulasan atau *review* yang diberikan oleh pengguna *Flip* melalui situs *Google Play Store*. Hasil penelitian menunjukkan bahwa akurasi yang dihasilkan sebesar 76,68%. Tahapan praproses data perlu diperbaiki sehingga dapat mengatasi kesalahan klasifikasi akibat penulisan komentar pengguna yang tidak sesuai dengan kaidah bahasa yang baik.

Penelitian yang dilakukan oleh Aluisius Dwiki Adhi Putra dan Safitri Juanita pada tahun 2021, dengan judul "*Analisis Sentimen Pada Ulasan Pengguna*

Aplikasi Bibit Dan Bareksa Dengan Algoritma KNN" (Adhi Putra, 2021). Penelitian ini bertujuan untuk menganalisis sentimen data ulasan atau *review* yang diberikan oleh pengguna *Bibit* dan *Bareksa* melalui situs *Google Play Store*. Hasil penelitian menunjukkan bahwa nilai *accuracy, precision,* dan *recall* yang dihasilkan sebesar 85,14%, 91,91%, dan 76,44% untuk *Bibit*, sedangkan untuk *Bareksa* yaitu 81,70%, 87,15%, 75,73%. Pada aplikasi *Bareksa*, opini yang didapatkan untuk setiap kelas menghasilkan di antaranya pada kelas negatif yaitu kurangnya pilihan pembayaran yang tersedia, sulitnya dalam melakukan registrasi terutama dalam verifikasi akun yang memakan cukup banyak waktu dan ketidakbermanfaatan aplikasi setelah di-*update* ke versi yang baru.

Penelitian yang dilakukan oleh Arief Asro'i dan Herny Februariyanti pada tahun 2022, dengan judul "*Analisis Sentimen Pengguna Twitter terhadap Perpanjangan PPKM Menggunakan Metode K-Nearest Neighbor*" (Asro'i & Februariyanti, 2022). Penelitian ini bertujuan untuk menganalisis sentimen *tweet* dari *twitter* tentang Perpanjangan PPKM. Hasil penelitian menunjukkan nilai akurasi sebesar 69,5%, recall 69,5%, dan presisi 68,7%. Hasil yang kurang memuaskan karena kita membutuhkan akurasi yang tinggi agar valid jika digunakan untuk evaluasi kebijakan.

Penelitian yang dilakukan oleh Abdy Yoga Syantara, Evi Dwi Wahyuni, dan Vinna Rahmayanti Setyaning Nastiti pada tahun 2021, dengan judul "*Analisis Sentimen Pada Media Sosial Twitter Menggunakan Naïve Bayes Classifier Terhadap Kata Kunci "#Asiangames2018"*" (Yoga Syantara dkk., 2021). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet*

dengan *keyword* "Asiangames2018". Hasil penelitian menunjukkan nilai akurasi sebesar 80,30% dan nilai positif 27, negatif 10, dan netral 16. Dan terakhir dari hasil pengujian nilai *K* dengan *range* 3 –5 didapatkan hasil paling optimal dan akurasi paling tinggi pada nilai *K* 5 dengan akurasi 80,30%, di bawahnya terdapat nilai *K* 4 dengan akurasi 67,07%, dan yang paling kecil dengan nilai *K* 3 dengan akurasi 49,09%. Dengan ini, untuk kedepannya diharapkan penelitian lain dapat menambahkan metode yang lebih serasi untuk disandingkan dengan klasifikasi *sentiment*, seperti contohnya metode *Social Network Analysis* yang bisa diterapkan pada studi kasus media sosial yang sama.

Penelitian yang dilakukan oleh Primandani Arsi, Bagus Adhi Kusuma, dan Azizan Nurhakim pada tahun 2021, dengan judul "*Analisis Sentimen Pindah Ibu Kota Berbasis Naive Bayes Classifier*" (Arsi dkk., 2021). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet* pada topik pemindahan ibukota Indonesia dengan cara mengklasifikasikan menjadi positif dan negatif. Hasil penelitian menunjukkan nilai akurasi sebesar 94,33%, *precision* sebesar 0,87 dan *recall* 0,99. Nilai harmonik *recall* dan *precision* atau yang disebut juga sebagai *F1-Score* adalah 0,92 yang berarti sistem ini sudah baik dalam mendeteksi sentimen.

Penelitian yang dilakukan oleh Elisa Febriyani dan Herny Februariyanti pada tahun 2023, dengan judul "*Analisis Sentimen Pindah Ibu Kota Berbasis Naive Bayes Classifier*" (Febriyani & Februariyanti, t.t.). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet* pada pada kata kunci "kampus merdeka" atau "merdeka belajar kampus merdeka" dalam unggahan, teks berbahasa

Indonesia, dan mengandung opini. Hasil penelitian menunjukkan pengklasifikasian 501 data opini *public* terhadap program kampus merdeka dengan *Naïve Bayes Classifier* mendapatkan hasil klasifikasi sentimen positif sebanyak 272 opini dan sentimen negatif sebanyak 229 opini dengan akurasi sebesar 57%. Dari hasil akurasi tersebut menjelaskan bahwa tingkat akurasi yang dicoba dengan pemodelan *Naïve Bayes* cukup kurang. Sedangkan evaluasi terhadap pemodelan klasifikasi yang diimplementasikan pada *k-fold cross validation* dengan membagi dataset menjadi 5-*fold* mendapatkan rata-rata akurasi 60%, presisi 64%, *recall* 58%, dan *f1-score* 58% yang berarti hasil evaluasi dengan *k-fold* mendapatkan jumlah akurasi lebih tinggi.

Penelitian yang dilakukan oleh I Komang Andi Sugiarta, Putri Anugrah Cahya Dewi, dan Nengah Widya Utami pada tahun 2023, dengan judul "*Analisa Sentimen Mahawiswa Terhadap Layanan STMIK Primakara Menggunakan Algoritma Naïve Bayes Dan K-Nearet Neighbor*" (I Komang Andi Sugiarta dkk., 2023). Penelitian ini bertujuan untuk menganalisis data *selection* yang diperoleh dari database PPM STMIK Primakara dari rentan tahun 2019-2022. Hasil penelitian menunjukkan pengklasifikasian 2608 data dengan *Naïve Bayes Classifier* mendapatkan hasil akurasi sebesar 64.37%. Sedangkan evaluasi terhadap pemodelan *K-Nearest Neighbor* mendapatkan rata-rata akurasi 85.06%.

Penelitian yang dilakukan oleh Antonius Mbay Ndapamuri, Danny Manongga, dan Ade Iriani pada tahun 2023, dengan judul "*Analisis Sentimen Ulasan Aplikasi Tripadvisor Dengan Metode Support Vector Machine, K-Nearest Neighbor, Dan Naive Bayes*" (Ndapamuri dkk., 2023). Penelitian ini bertujuan

untuk menganalisis 1000 data ulasan mengenai aplikasi Tripadvisor yang diperoleh dari database *Google Play Store* periode 7 Januari 2018 sampai dengan 28 Februari 2023. Hasil penelitian pada pengujian evaluasi model klasifikasi sentimen, diperoleh hasil dengan akurasi menggunakan model *Support Vector Machine* sebesar 89,8%, model *K-Nearest Neighbor* sebesar 89,02%, dan *Naïve Bayes* sebesar 88,65%.

Penelitian yang dilakukan oleh Jepi Supriyanto, Debby Alita, dan Auliya Rahman Isnain pada tahun 2023, dengan judul "*Penerapan Algoritma K-Nearest Neighbor (K-NN) Untuk Analisis Sentimen Publik Terhadap Pembelajaran Daring*" (Supriyanto dkk., t.t.). Penelitian ini bertujuan untuk menganalisis dataset *tweet* 1825 dari *twitter*. Hasil penelitian menunjukkan pengklasifikasian *K-Nearest Neighbor* mendapatkan rata-rata akurasi 84,93%.

Penelitian yang dilakukan oleh Rizqi Agung Permana dan Sucitra Sahara pada tahun 2023, dengan judul "*Algoritma K-Nearest Neighbor Pada Analisa Sentimen Review Produk Router*" (Permana & Sahara, 2023). Penelitian ini bertujuan untuk menganalisis dataset *dari sebuah situs e-commerce layaknya website www.amazon.com (review produk), www.reviewcentre.com* pengklasifikasian *K-Nearest Neighbor* mendapatkan rata-rata akurasi 83.20%.

Penelitian yang dilakukan oleh Indra Febriansyah, Muhammad Fikry dan Yusra pada tahun 2023, dengan judul "*Analisis Sentiment di Twitter terhadap Anies Baswedan sebagai Bakal Calon Presiden 2024 Menggunakan Metode K-Nearest Neighbor*" (Febriansyah dkk., 2023). Penelitian ini bertujuan untuk menganalisis sentimen di *twitter* dari data *tweet* dengan *keyword* "Anies Calon Presiden 2024".

Hasil penelitian ini menunjukkan bahwa metode K-Nearest Neighbor berhasil diterapkan dalam analisis sentimen di Twitter terhadap Anies Baswedan sebagai bakal calon presiden 2024. Model K-Nearest Neighbor dengan performa terbaik terdapat pada fold pertama dengan kombinasi parameter nilai k = 14 dan nilai threshold = 12 pada Document Frequency (DF), dengan akurasi mencapai 87,35%, presisi 87,39%, recall 85,3%, dan f1 score 86,13%.

Penelitian yang dilakukan oleh Indra Yumarlin MZ, Jemmy Edwin Bororing, Sri Rahayu, dan Jeffry Andhika Putra pada tahun 2023, dengan judul "*Analisis Sentimen Pengguna Apliaksi Shopee Menggunakan Metode Naive Bayes Classifier dan K-NN*" (MZ dkk., 2023). Penelitian ini bertujuan untuk menganalisis sentimen ulasan di *Google Play Store*. Hasil penelitian ini menunjukkan dari penerapan metode *Naïve Bayes Classifier* untuk data ulasan positif *machine learning* memprediksi komentar positif sebesar 742, komentar negatifnya 401 dan netral sebesar 6, dengan nilai *recall* sebesar 64.58% dan *precision* sebesar 98.76. Untuk prediksi ulasan negatif *machine learning* memprediksi komentar positif sebesar 10, komentar negative 519 dan netralnya sebesar 9, dengan nilai *recall* sebesar 96.74%, *precision* sebesar 56.41%, dan nilai *accuracy* yang di dapat sebesar 75.97%. Sedangkan penerapan metode *K- Nearest Neighbor* menunjukkan untuk data ulasan positif dari *machine learning* memprediksi komentar positif sebesar 154, komentar negatifnya 39 dan netral sebesar 956, dengan nilai *recall* sebesar 13.40% dan *precision* sebesar 69.3%. Untuk ulasan negatif *machine learning* prediksi komentar positifsebesar 56, komentar negative 80 dan netralnya sebesar

420, dengan nilai *recall* sebesar 14.87%, *precision* sebesar 61.07%, dan nilai *accuracy* yang di dapat sebesar 16,69%.

Penelitian yang dilakukan oleh Egi Fransisco Saputra dan Muhammad Rizky Pribadi pada tahun 2023, dengan judul "*Analisis Sentimen Komentar Pada Kanal Youtube The Lazy Monday Menggunakan Algoritma Naive Bayes*" (Saputra & Pribadi, 2023). Penelitian ini bertujuan untuk menganalisis sentimen ulasan di *Google Play Store*. Hasil penelitian ini menunjukkan pada pengujian pertama hanya mendapatkan tingkat akurasi sebesar 45% dengan *precission* positf 42%, negatif 36%, dan 58%, *recall* positif 50%, negatif 57% dan netral 35%. Pada pengujian atau skenario kedua dengan menggunakan data *imbalance* yang sama namun diiringi dengan metode *oversampling* SMOTE dengan melakukan *K-fold Cross Validation* untuk mendapatkan *best validation* dapat dihasilkan rata-rata akurasi yang diapatkan lebih baik yakni sebesar 67% *precission* positf 62%, negatif 67 % dan 76%, *recall* positif 71%, negatif 93%, dan netral 35%.

2.2. Keaslian Penelitian

Table 2.1. Matriks literatur review dan posisi penelitian
Analisis Sentimen Publik Terhadap Elektabilitas Ganjar Pranowo Di Tahun Politik 2024 Di Twitter
Dengan Algoritma KNN Dan Naïve Bayes

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
1	*Sentiment Analysis about E-Commerce from Tweets Using Decision Tree, K-Nearest Neighbor, and Naïve Bayes*	Achmad Bayhaqy, Sfenrianto Sfenrianto, Kaman Nainggolan, Emil R. Kaburuan. International Conference on Orange Technologies (ICOT), 06 May 2019, pp. 1-6.	Membandingkan *result* untuk mengetahui *classifier* yang memberikan hasil terbaik dalam hal rasio *recall* dan akurasi menggunakan *data mining*.	Pengklasifikasi Naïve Bayes adalah pengklasifikasi terbaik untuk digunakan dengan kumpulan data media sosial karena menyediakan lebih akurat dan prediksi yang tepat.	Menggunakan yang lebih besar dan lebih kompleks dataset dengan jumlah label yang lebih banyak dan jangkauan *e-commerce* yang lebih banyak dan dapat mencakup bahasa Indonesia non-standar.	Penelitian sebelumnya, menggunakan *Decision Tree, KNN,* dan *Naïve Bayes* untuk menganalisis sentimen tentang *E-Commerce* di *Twitter*. Penelitian berikutnya menggunakan *KNN* dan *Naïve Bayes* untuk menganalisis sentimen terhadap Elektabilitas Ganjar Pranowo di Tahun Politik 2024 di *twitter*.
2	Penerapan Analisis Sentimen Pada Pengguna *Twitter* Menggunakan Metode *K-Nearest Neighbor*	Akhmad Deviyanto, Muhammad Didik Rohmad Wahyudi. JISKa (Jurnal Informatika Sunan Kalijaga), Vol. 3, No. 1, MEI, 2018, Pp. 1–13.	Menganalisis sentimen di *twitter* dari data *tweet* yang mengandung kata 'AHY' dari mulai tanggal 1 Januari 2017 sampai dengan 31 Januari 2017.	Penelitian tentang analisis sentimen pengguna *Twitter* terhadap topik Pilkada DKI 2017 dengan menggunakan metode *K-Nearest Neighbor* telah berhasil dilakukan. Hasil akurasi terbesar adalah 67,2% dengan nilai k=5. Sedangkan nilai presisi tertinggi sebesar 56,94% saat k=5 dan *recall* terbesar 78,24 % ketika k=15.	Akurasi terbesar dengan metode *K-Nearest Neighbor* adalah 67,2% dengan nilai *K*=5. Penelitian selanjutnya akan lebih baik jika dilakukan atau diuji menggunakan dua metode atau algoritma dalam mengolah data.	Penelitian sebelumnya, menggunakan *K-Nearest Neighbor* untuk menganalisis sentimen tentang *Pilkada DKI 2017* di *Twitter*. Penelitian berikutnya menggunakan *KNN* dan *Naïve Bayes* untuk menganalisis sentimen terhadap Elektabilitas Ganjar Pranowo di Tahun Politik 2024 di *twitter*.

Table 2.1. (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
3	Analisis Sentimen Pengguna *Twitter* Terhadap Polemik Persepakbolaan Indonesia Menggunakan Pembobotan *TF-IDF* dan *K-Nearest Neighbor*	J. A. Septian, T. M. Fachrudin, dan A. Nugroho. INSYST, vol. 1, no. 1, pp. 43-49, Aug. 2019.	Menganalisis sentimen di *twitter* dari data *tweet* dengan *query* '@PSII' dari mulai tanggal 21 Februari 2019 sampai dengan 4 Mei 2019.	Dari 2000 data *tweet* terkait polemik persepakbolaan Indonesia yang diambil dari akun twitter PSSI dengan proporsi 790 *tweet* positif dan 1210 *tweet* negatif, dilakukan percobaan untuk mencari model KNN dengan akurasi terbaik menggunakan range nilai k=1 hingga k=30 yang adalah bilangan ganjil, didapatkan akurasi optimal pada k=23 dengan akurasi sebesar 79,99% dan *error rate* sebesar 20,01%.	Dari 10 *tweet* baru yang dilakukan pengujian untuk mendapatkan sentimen menyisakan 1 *tweet* yang tidak sesuai perdiksi. Penelitian selanjutnya akan lebih baik jika memberikan nilai *K* yang bervariatif dalam mengolah data.	Penelitian sebelumnya, menggunakan *TF-IDF* dan *K-Nearest Neighbor* untuk menganalisis sentimen tentang *Polemik Pesepakbolaan Indonesia* di *Twitter*. Penelitian berikutnya menggunakan *KNN* dan *Naïve Bayes* untuk menganalisis sentimen terhadap Elektabilitas Ganjar Pranowo di Tahun Politik 2024 di *twitter*.
4	Analisis Sentimen Pengguna *Twitter* Terhadap Dompet Elektronik Dengan Metode *Lexicon Based* Dan *K-Nearest Neighbor*	Siti Saidah Salim dan Joanna Mayary. Jurnal Ilmiah Informatika Komputer, Vol 25, No. 1 (2020).	Menganalisis sentimen di *twitter* dari data *tweet* dengan *query* 'OVO', 'Gopay', dan 'LinkAja'.	Pengujian menggunakan *confusion matrix* memperoleh hasil akurasi dari masing-masing data menggunakan metode *K-Nearest Neighbor* untuk tweet OVO cenderung positif ditunjukkan oleh nilai akurasi data 86,91% dan kecenderungan negatif *tweet* OVO memiliki nilai akurasi data 13,09%, tweet *Gopay* cenderung positif dengan akurasi data 94,05% dan kecenderungan negatif *tweet* Gopay memiliki nilai akurasi data 5,95%, serta *tweet* LinkAja cenderung positif dengan akurasi	Aplikasi dapat dikembangkan dengan penambahan fitur *real-time* pada visualisasi data, sehingga dapat diakses kapan saja. Pemilihan tingkat ketepatan dalam melakukan prediksi, membutuhkan konsultasi dengan pakar bahasa untuk mengatasi data uji yang memiliki kalimat ambigu.	Penelitian sebelumnya, menggunakan *Lexicon Based* dan *K-Nearest Neighbor* untuk menganalisis sentimen tentang *Dompet Elektronik* di *Twitter*. Penelitian berikutnya menggunakan *KNN* dan *Naïve Bayes* untuk menganalisis sentimen terhadap Elektabilitas Ganjar Pranowo di Tahun Politik 2024 di *twitter*.

Table 2.1. (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
				data 76,31% dan kecenderungan negatif *tweet* LinkAja memiliki nilai akurasi data 23,69%.		
5	Analisis Sentimen Kinerja Dewan Perwakilan Rakyat (DPR) Pada Twitter Menggunakan Metode *Naïve Bayes Classifier*	Dianati Duci Putri, Gigih Forda Nama, dan Wahyu Eko Sulistiono. Jurnal Informatika dan Teknik Elektro Terapan (JITET), Vol. 10 No. 1, Januari 2022.	Menganalisis sentimen di *twitter* dari data *tweet* dengan *query* 'DPR'.	*Accuracy score* sebesar 0.8 atau 80% hal ini berarti sistem mampu memprediksi 80% secara akurat dari total *data testing* sebesar 20%. Berdasarkan hasil analisis sistem mendapatkan klasifikasi *tweet* dari *twitter* mengenai DPR sebanyak 95 positif, 693 netral dan 758 negatif dari data hasil *crawling* sebanyak 1546.	Sistem *sentiment analysis* ini sebaiknya menggunakan *database* agar mampu menyimpan data dalam jumlah besar. Visualisasi hasil analisis sebaiknya menggunakan fitur filter pada bagian tabel berdasarkan tanggal *crawling* sehingga mempermudah untuk melihat *sentiment analysis*.	Penelitian sebelumnya, menggunakan *Naïve Bayes Classifier* untuk menganalisis sentimen tentang *Kinerja Dewan Perwakilan Rakyat (DPR) Pada Twitter*. Penelitian berikutnya menggunakan *KNN* dan *Naïve Bayes* untuk menganalisis sentimen terhadap Elektabilitas Ganjar Pranowo di Tahun Politik 2024 di *twitter*.
6	Penerapan Algoritma *Naïve Bayes* untuk Analisis Sentimen *Review* Data *Twitter* BMKG Nasional	Dedi Darwis, Nery Siskawati, Zaenal Abidin. Jurnal Tekno Kompak, Vol 15, No. 1 (2021).	Menganalisis sentimen di *twitter* dari data *tweet* dengan *query* 'BMKG'.	Proses ekstrakasi data dari *Twitter* BMKG Nasional menggunakan Bahasa pemrograman *Python 3.74* dengan tahapan *Preprocessing* yang meliputi *casefolding*, *filtering*, tokenisasi, *slang replacement* dan *stopword removal*. Tingkat akurasi berdasarkan pengujian yang dilakukan adalah 68,97%.	Gunakan lebih dari satu algoritma dengan studi kasus yang lebih luas dalam skala nasional agar dapat memengaruhi tingkat akurasi yang lebih baik.	Penelitian sebelumnya, menggunakan *Naïve Bayes* untuk menganalisis sentimen tentang *BMKG* di *Twitter*. Penelitian berikutnya menggunakan *KNN* dan *Naïve Bayes* untuk menganalisis sentimen terhadap Elektabilitas Ganjar Pranowo di Tahun Politik 2024 di *twitter*.

Table 2.1. (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
7	Implementasi Algoritma *Naïve Bayes* Terhadap Analisis Sentimen Opini Film Pada *Twitter*	Fajar Ratnawati. Jurnal Inovtek Polbeng – Seri Informatika, Vol. 3, No. 1, Juni 2018.	Menganalisis sentimen di *twitter* dari data *tweet* dengan *hashtag* '#judul film' dari mulai tanggal 1 Januari 2015 sampai dengan 31 Desember 2015.	Semakin banyak *data training* yang digunakan maka akan memengaruhi kinerja dari sistem. Hasil akurasi akan semakin tinggi dan itu menandakan sistem berhasil melakukan klasifikasi dengan baik. Akurasi tertinggi didapat pada *fold* kedua yaitu 90%, *precision* 92%, *recall* 90% dan *f-measure* 90%.	Terdapat kesalahan dalam proses klasifikasi sistem yang disebabkan oleh penggunaan dataset yang kurang tepat. Keadaan ini yang menyebabkan banyak ditemukan fitur kemunculan data yang bukan termasuk kategorinya pada data uji yang digunakan. Contohnya dataset yang digunakan dalam data positif terdapat kata "lucu", kemudian pada data negatif juga terdapat kata "lucu" yang berasal dari kata "tidak lucu". Kata yang sama pada dataset positif dan negatif inilah yang mengakibatkan kesalahan dalam proses klasifikasi data.	Penelitian sebelumnya, menggunakan *Naïve Bayes* untuk menganalisis sentimen pada *Opini Film* di *Twitter*. Penelitian berikutnya menggunakan *KNN* dan *Naïve Bayes* untuk menganalisis sentimen terhadap Elektabilitas Ganjar Pranowo di Tahun Politik 2024 di *twitter*.
8	Analisis Sentimen Opini Masyarakat Indonesia Terhadap Kebijakan PPKM	Taofik Krisdiyanto dan Erry Maricha Oki Nurharyanto.	Menganalisis sentimen di *twitter* dari data *tweet*	Analisis sentimen yang sudah dilakukan menggunakan *software Rstudio* terhadap data kata PPKM pada *twitter* yang berjumlah 1000	Diperbanyak jumlah kata yang terdaftar dalam *library* sehingga	Penelitian sebelumnya, menggunakan *Naïve Bayes Classifier* untuk menganalisis sentimen terhadap *Kebijakan*

Table 2.1. (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
	Pada Media Sosial *Twitter* Menggunakan *Naïve Bayes Classifier*	Jurnal CoreIT, Vol 7, No. 1 (2021).	dengan *hashtag* 'PPKM'dengan metode *crawling* secara otomatis menggunakan *software Rstudio.*	data ulasan, diperoleh hasil yaitu berupa sentimen positif sebanyak 99% dan sentimen negatif sebanyak 1% atau terlihat dari hasil klasifikasi emosi memiliki mayoritas "*unknown*" atau tidak diketahui yang sebagian besar kata tersebut memiliki polaritas positif.	meningkatkan keakuratan klasifikasi.	*PPKM* di *Twitter*. Penelitian berikutnya menggunakan *KNN* dan *Naïve Bayes* untuk menganalisis sentimen terhadap Elektabilitas Ganjar Pranowo di Tahun Politik 2024 di *twitter*.
9	Analisis Sentimen Program Acara Di SCTV Pada *Twitter* Menggunakan Metode *Naïve Bayes* Dan *Support Vector Machine*	Dery Anjas Ramadhan dan Erwin Budi Setiawan. E-Proceeding of Engineering: Vol 6, No 2 (2019).	Menganalisis sentimen di *twitter* dari data *tweet* yang mengandung kata "ftv, sinetron, entertainmentsctv, dan liputan6sctv" dari mulai awal Mei 2019 sampai dengan akhir Juni 2019.	Akurasi terbaik antara metode *Naïve Bayes* dan *Support Vector Machine* adalah Metode *Support Vector Machine* dengan Seluruh Program Acara didapatkan hasil akurasi 88,57% berikut perkategori yang diperoleh, Berita didapatkan hasil akurasi 79,81%, Entertaiment didapatkan hasil akurasi 89,80%, Sinetron didapatkan hasil akurasi 73,68% dan FTV didapatkan hasil akurasi 87,74%.	Harus menambahkan Fitur tambahan dalam pengolahan data seperti *Tweed – Based dan TF – IDF* untuk memvariasikan pengolahan datanya agar mendapatkan perbandingan akurasi yang lebih baik lagi. Membuat *Korpus* dan *Stopword* yang lebih baik lagi agar dalam melakukan pembuatan data tidak terjadi kehilangan kata – kata penting saat *preprocessing*.	Penelitian sebelumnya, menggunakan *Naïve Bayes* dan *Support Vector Machine* untuk menganalisis sentimen terhadap *Program Acara Di SCTV* di *Twitter*. Penelitian berikutnya menggunakan *KNN* dan *Naïve Bayes* untuk menganalisis sentimen terhadap Elektabilitas Ganjar Pranowo di Tahun Politik 2024 di *twitter*.

2.3. Landasan Teori

2.3.1. Analisis Sentimen

Analisis Sentimen merupakan teknik atau proses analisis teks secara *online* berupa opini atau pendapat secara publik yang terdiri dari polaritas sentimen positif, negatif, atau pun netral. Analisis sentimen adalah proses mengekstraksi, memahami dan mengolah data berupa teks yang tidak terstruktur secara otomatis guna mendapatkan informasi sentimen yang terdapat pada sebuah kalimat pendapat atau opini (Arsi dkk., 2021). Dalam penelitian ini melakukan studi kasus terkait elektabilitas seorang Ganjar Pranowo diusung menjadi salah satu Calon Presiden di tahun 2024 mendatang.

2.3.2. *Twitter*

Twitter merupakan media sosial yang seringkali menjadi pusat *trending* mengenai isu di dunia skala nasioal maupun internasional yang dijadikan warganet sebagai media untuk menyuarakan opini terkait sentimen terhadap apa pun yang terkini diperbincangkan di jejaring sosial yang begitu kompleks. Pengguna *twitter* di Indonesia memiliki 19,5 juta pengguna total 500 juta pengguna global.

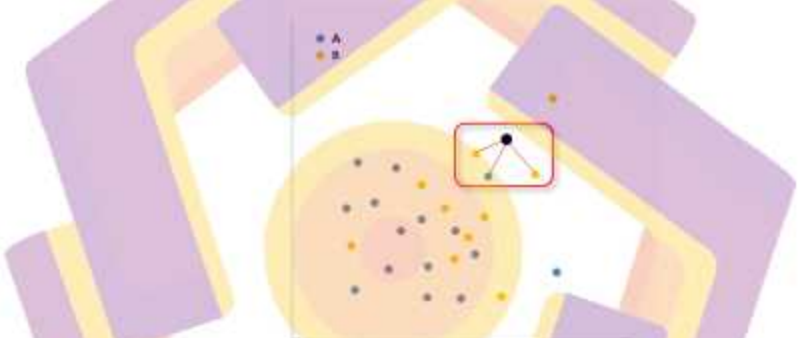
2.3.3. Algoritma *K-Nearest Neighbors* (KNN)

K-Nearst Neighbors (KNN) adalah suatu metode yang menggunakan algoritma *supervised* di mana hasil dari *query instance* yang baru diklasifikan berdasarkan mayoritas dari kategori pada KNN (Sikki, 2009). Tujuan dari algoritma KNN adalah untuk mengklasifikasi objek baru berdasarkan atribut dan *training*

samples. Di mana hasil dari sampel uji yang baru diklasifikasikan berdasarkan mayoritas dari kategori pada KNN(Krisandi dkk., 2013). Secara umum, rumus formula *euclidean distance* pada *K-Nearest Neighbors* adalah sebagai berikut:

$$dis\ (x_1, x_2) = \sqrt{\sum_{i=0}^{n}(x_{1i} - x_{2i})^2}$$

$$dis\ (x_1, x_2) = \sqrt{\sum_{i=0}^{n}(x_{1i} - x_{2i})^2 + (y_{1i} - y_{2i})^2 + naïve}$$



Gambar 2.1. Visualisasi *K-Nearest Negihbors*

Gambar 2.1 di atas menunjukkan adanya data dari dua kelas, yaitu A (biru), B (kuning), dan data baru berwarna hitam yang digunakan untuk memprediksi kelasnya dengan *K-Nearest Neighbors*. Tiga data *point* yang di-*highlight* bersama data baru berwarna hitam menunjukkan bahwa nilai *k* yang digunakan adalah 3.

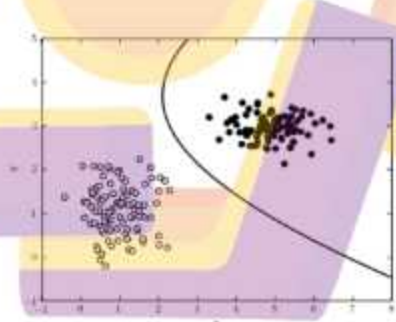
2.3.4. Algoritma *Naïve Bayes*

Naïve Bayes dikemukakan oleh ilmuwan Inggris Thomas Bayes, yaitu memprediksi peluang di masa depan berdasarkan pengalaman di masa sebelumnya

sehingga disebut sebagai Teorema Bayes (Hayuningtyas & Sari, 2019). *Naïve Bayes* untuk setiap kelas keputusan menghitung probabilitas dengan syarat bahwa kelas keputusan adalah benar, mengingat *vector* informasi objek. Algoritma ini mengasumsikan bahwa atribut objek adalah independen. Probabilitas yang terlibat dalam memproduksi perkiraan akhir dihitung sebagai jumlah frekuensi dari "master" tabel keputusan. Secara umum, rumus pada *Naïve Bayes* adalah sebagai berikut:

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

A, B	= *events*
P(A|B)	= *probability of A given B is true*
P(B|A)	= *probability of B given A is true*
P(A), P(B)	= *the independent probabilities of A and B*



Gambar 2.2. Visualisasi *Naïve Bayes*

Pada Gambar 2.2 di atas menunjukkan tahapan penelitian, peneliti menerapkan dua metodologi pengujian sebagai perbandingan suatu hasil setelah melakukan *preprocessing* dan pembagian data atau *split data* antara data latih dan

data uji. Dua metodelogi pengujian yang digunakan ialah algoritma *K-Nearest Neighbors* (KNN) dan *Naïve Bayes*.

2.3.5. *Confusion Matrix*

Confusion Matrix merupakan metode untuk memprediksi perhitungan *matrix* pada akurasi dalam *data mining*. Sebagai evaluasi model, *confusion matrix* terdiri dari *data test* yang telah diprediksi antara benar dan tidak benar terhadap pemodelan klasifikasi data yang telah diujikan. Selain *confusion matrix*, penulis juga menghitung *accuracy, precision, recal,* dan *f-1 score* dengan rumus sebagai berikut:

$$Accuracy = \frac{Number\ of\ correctly\ classified\ classes}{Total\ number\ of\ classes} \times 100\%$$

$$Precission = \frac{True\ Positive}{True\ Positive + False\ Positive} \times 100\%$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \times 100\%$$

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

2.3.6. *Python*

Python adalah bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. *Python* diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif. *Python* juga didukung oleh komunitas yang besar(Syahrudin, 2018).

2.3.7. *RapidMiner*

RapidMiner adalah media interaktif pengguna untuk melakukan pembelajaran mesin dan proses penambangan data. *RapidMiner* bersifat *open-source* atau bersifat umum dan dapat dimodifikasi oleh penggunanya. Mewakili pendekatan modular untuk merancang bahkan untuk masalah yang sangat kompleks, di mana konsep operator modular yang memungkinkan desain rantai operator bersarang kompleks untuk masalah pembelajaran yang besar. *RapidMiner* menggunakan XML untuk menggambarkan proses-proses penemuan pengetahuan pemodelan pohon operator. *RapidMiner* memiliki operator yang mudah dipahami untuk input dan output data dalam format file yang berbeda, dan berisi lebih dari 100 skema pembelajaran untuk tugas klasifikasi, regresi, dan pengelompokan (Naik & Samant, 2016).

BAB III

METODE PENELITIAN

3.1. Jenis, Sifat, dan Pendekatan Penelitian

3.1.1. Jenis Penelitian

Jenis penelitian yang akan dilakukan oleh peneliti, yaitu penelitian kuantitatif yang berhubungan dengan sentimen publik di *twitter* menggunakan algoritma *K-Nearest Neighbors* (KNN) dan *Naïve Bayes*. *Output* dari penelitian ini yaitu berupa data sentimen publik dan komparasi akurasi.

3.1.2. Sifat Penelitian

Pada penelitian ini bersifat eksperimen. Peneliti nantinya menerapkan beberapa teknik untuk mendapatkan akurasi yang terbaik, terhadap dataset serta model algoritma *K-Nearest Neighbors* (KNN) dan *Naïve Bayes* yang akan diuji.

3.1.3. Pendekatan Penelitian

Pada penelitian ini, peneliti menggunakan pendekatan penelitian kuantitatif dengan studi eksperimen, yaitu peneliti melakukan eksperimen terhadap dataset yang telah ada untuk kemudian diuji menggunakan algoritma *K-Nearest Neighbors* (KNN) dan *Naïve Bayes* yang tujuannya untuk meningkatkan komparasi hasil akurasi. Selanjutnya, dokumentasi atas hasil evaluasi dijabarkan dalam bentuk statistik, tabel, serta kesimpulan.

3.2. Metode Pengumpulan Data

Metode pengumpulan data yang dilakukan peneliti, yaitu dengan melakukan studi literatur dan dokumen. Data yang digunakan dalam penelitian ini adalah data *crawling* dari media sosial *twitter* dengan *query* "Ganjar Pranowo Capres". Dataset yang diuji adalah dataset publik.

Dataset yang dikumpulkan adalah dataset yang telah melalui proses *preprocessing* berupa data *tweet* dari *twitter* mulai dari 1 Januari 2023 sampai dengan 12 Mei 2023 sebanyak 2000 *tweet*.

3.3. Metode Analisis Data

Metode analisis data yang digunakan peneliti, yaitu analisis deskriptif kuantitatif untuk melihat dan mengetahui performa data pada masa lalu, agar dapat diambil kesimpulan dari hal tersebut. Metode analisis data jenis ini diaplikasikan pada data dengan *volume* besar yang pada akhirnya akan menghasilkan *output* sebagai bahan klasifikasi teks. Oleh karena itu, untuk mendapatkan data yang berkualitas, digunakan beberapa teknik pengolahan data sebagai berikut:

a. *Data validation*

   Digunakan untuk mengidentifikasikan dan menghapus data yang ganjil (*outlier/noise*), data yang tidak konsisten, dan data yang tidak lengkap (*missing value*).

b. *Data integration and transformation*

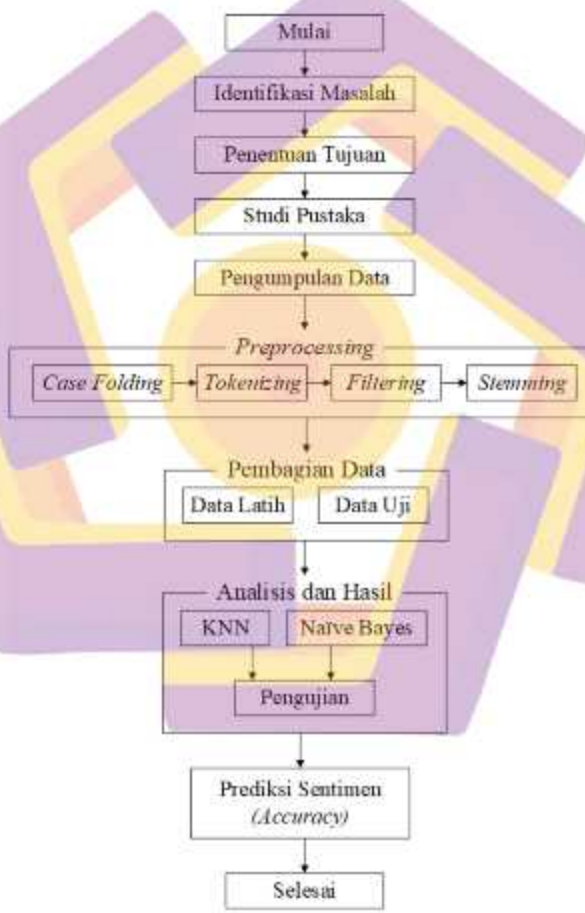
Digunakan untuk meningkatkan akurasi dan efisiensi algoritma. Data yang digunakan dalam penelitian ini adalah data yang berlabel. Data ditransformasikan ke dalam aplikasi *RapidMiner*.

c. *Data size reduction and discritization*

Digunakan untuk memperoleh dataset dengan jumlah *atribute* dan *record* yang lebih sedikit, tetapi bersifat informatif.

3.4. Alur Penelitian

Pada penelitian ini berisi diagram alur langkah penelitian secara lengkap dan terinci, yaitu algoritma yang digunakan, *rute*, pemodelan, desain, yang terkait dengan masalah, atau aspek dari perancangan sistem. Berikut gambar *flowchart* penelitian di bawah ini:



Gambar 3.1 *Flowchart* Penelitian

Keterangan alur *Flowchart* Penelitian di atas:

a) Penulis melakukan Identifikasi Masalah.

b) Penulis menentukan Tujuan penelitian.

c) Penulis melakukan Studi Pustaka.

d) Penulis melakukan Pengumpulan Data dengan *crawling* data teks.

e) Penulis melakukan *preprocessing* dataset mentah dengan *case folding*, *tokenizing*, *filtering*, dan *stemming*.

f) Dataset yang telah melalui proses *preprocessing* akan dibagi menjadi Data Latih dan Data Uji sebagai analisis data.

g) Setelah membagi dataset untuk dilatih dan diujikan, kemudian dilakukan Pengujian dengan algoritma *K-Nearest Neighbors* dan *Naïve* Bayes.

h) Pengujian dari kedua algoritma tersebut akan menghasilkan tingkat akurasi untuk dikomparasi.

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

4.1. Analisis Kebutuhan Data

Tahap analisis kebutuhan data merupakan salah satu proses penting dalam penelitian ini, kebutuhan data yang dimaksud adalah kebutuhan data yang akan digunakan dalam proses *training* model *machine learning* yang akan dibangun. Pada penelitian ini data yang dibutuhkan adalah berupa dataset *crawling* melalui perangkat lunak *RapidMiner* untuk melakukan tarik data berupa *tweet* dengan *query* "Ganjar Pranowo Capres" sebanyak 2000 *tweet*. Pada tahap ini, *labeling* sentimen terdiri dari dua kelas yaitu positif dan negatif.

4.2. *Preprocessing*

Pada tahap *Preprocessing*, penelitian ini melakukan serangkaian tahapan berupa *case folding*, *tokenizing*, *filtering* (*stopwords removal*), dan *stemming*. Berikut hasil *import* beberapa *sample* dataset tersebut ke dalam *Google Colab* dapat dilihat pada Tabel 4.1 berikut ini:

Tabel 4.1. Dataset *tweet* "Ganjar Pranowo Capres"

	Tweet
0	Kemiskinan Jateng hingga "Capres Boneka" Bikin...
1	#GanjarMenangTotal @ganjarpranowo Disambut War...
2	Bakal Capres Ganjar Pranowo nampaknya mulai ge...
3	Ribuan ulama deklarasikan dukung Ganjar Pranow...
4	@KakekHalal: Ribuan Warga Manado Sambut Ganja...

Pada Tabel 4.1 di atas menunjukkan *import* dataset yang masih mentah dan belum dilakukan *preprocessing* data.

4.2.1. *Case Folding*

Case Folding pada *DataFrame tweet* di atas menggunakan fungsi *lower()* pada *class Series.str library Pandas*. Penggunakan Fungsi internal *pandas* ini jauh lebih cepat dibandingkan jika kita melakukan iterasi untuk semua *row* pada *DataFrame* dan melakukan *Case Folding row by row* menggunakan *lower()* biasa. *Listing source code* dari *case folding* dapat dilihat pada Tabel 4.2 berikut:

Tabel 4.2. *Source Code Case Folding*

No	Source Code
1	`TWEET_DATA['Tweet'] = TWEET_DATA['Tweet'].str.lower()`

2	`print('HASIL CASE FOLDING: \n')`
	`print(TWEET_DATA.head(5))`
	`print('\n\n\n')`

Keterangan kode program *Source Code Case Folding* di atas:

a) *Cell 1*, penggunaan fungsi *Series.str.lower()* pada *library* Pandas.

b) *Cell 2*, melihat atau menampilkan hasil dari *running* kode program pada *cell 1* di atas.

Hasilnya dapat dilihat pada Tabel 4.3 berikut ini:

Tabel 4.3. *Case Folding*

	Tweet
0	kemiskinan jateng hingga "capres boneka" bikin...
1	#ganjarmenangtotal @ganjarpranowo disambut war...
2	bakal capres ganjar pranowo nampaknya mulai ge...
3	ribuan ulama deklarasikan dukung ganjar pranow...
4	@kakekhalal: ribuan warga manado sambut ganja...

Pada Tabel 4.3 di atas menunjukkan hasil dari *case folding* yang mana akan mengubah semua huruf menjadi kecil.

4.2.2. *Tokenizing*

Tokenizing pada *DataFrame tweet* di atas menggunakan *library* dari *nltk.tokenize* untuk *word_tokenize* dan *nltk.probability* untuk *FreqDist*. *Listing source code* dari *case folding* dapat dilihat Tabel 4.4 berikut ini:

Tabel 4.4. *Source Code Tokenizing*

No	Source Code
1	`import string` `import re`

2	`from nltk.tokenize import word_tokenize`
	`from nltk.probability import FreqDist`
	`import nltk`
	`nltk.download('punkt')`

3	`def remove_tweet_special(text):` `  text = text.replace('\\t'," ").replace('\\n' " ").replace('\\u'," ").replace('\\',"")`

Tabel 4.4. (Lanjutan)

4	`text = text.encode('ascii', 'replace').decode('ascii')`
5	`text = ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)"," ", text).split())`
6	`return text.replace("http://", " ").replace("https://", " ")`
7	`TWEET_DATA['Tweet'] = TWEET_DATA['Tweet'].apply(remove_tweet_special)`
8	`def remove_number(text):` `  return re.sub(r"\d+", "", text)`
9	`TWEET_DATA['Tweet'] = TWEET_DATA['Tweet'].apply(remove_number)`
10	`def remove_punctuation(text):` `  return text.translate(str.maketrans("","",string punctuation))`
11	`TWEET_DATA['Tweet'] = TWEET_DATA['Tweet'].apply(remove_punctuation)`
12	`def remove_whitespace_LT(text):` `  return text.strip()`
13	`TWEET_DATA['Tweet'] = TWEET_DATA['Tweet'].apply(remove_whitespace_LT)`
14	`def remove_whitespace_multiple(text):` `  return re.sub('\s+',' ',text)`

Tabel 4.4. (Lanjutan)

15	`TWEET_DATA['Tweet'] = TWEET_DATA['Tweet'].apply(remove_whitespace_multiple)`
16	`def remove_singl_char(text):` `  return re.sub(r"\b[a-zA-Z]\b", "", text)`
17	`TWEET_DATA['Tweet'] = TWEET_DATA['Tweet'].apply(remove_singl_char)`
18	`def word_tokenize_wrapper(text):` `  return word_tokenize(text)`
19	`TWEET_DATA['tweet_tokens'] = TWEET_DATA['Tweet'].apply(word_tokenize_wrapper)`

19	`print('Tokenizing Result : \n')`
	`print(TWEET_DATA['tweet_tokens'].head())`
	`print('\n\n\n')`

Keterangan kode program *Source Code Tokenizing* di atas:

i) *Cell 1*, impor *library* dari *string* dan *re*.

j) *Cell 2*, impor *library word_tokenize* dan *FreqDist* dari NLTK.

k) *Cell 3*, hapus tab, baris baru, dan potongan belakang.

l) *Cell 4*, hapus non ASCII (emotikon, huruf cina, dan lain sebagainya).

m) *Cell 5*, hapus *mention*, tautan, dan tagar atau *hashtag*.

n) *Cell 6*, hapus URL yang tidak lengkap.

o) *Cell 7*, menerapkan semua proses di atas.

p) *Cell 8*, hapus nomor-nomor.

q) *Cell 9*, menerapkan proses penghapusan nomor-nomor.

r) *Cell 10*, hapus tanda baca.

s) *Cell 11*, menerapkan proses penghapusan tanda baca.

t) *Cell 12*, hapus spasi awal dan akhir.

u) *Cell 13*, menerapkan proses penghapusan spasi awal dan akhir.

v) *Cell 14*, hapus beberapa spasi menjadi satu spasi.

w) *Cell 15*, menerapkan proses penghapusan beberapa spasi menjadi satu spasi.

x) *Cell 16*, hapus *single char*.

y) *Cell 17*, menerapkan proses penghapusan *single char*.

z) *Cell 18*, memadukan semua tokenisasi dari *library* NLTK.

aa) *Cell 19*, menerapkan proses tokenisasi dari *library* NLTK.

bb) *Cell 20*, melihat atau menampilkan hasil dari *running* kode program.

Hasilnya dapat dilihat pada Tabel 4.5 berikut ini:

Tabel 4.5. *Tokenizing*

	Tweet
0	[kemiskinan, jateng, hingga, capres, boneka, b...
1	[disambut, warga, dan, tokoh, gereja, yang, te...
2	[bakal, capres, ganjar, pranowo, nampaknya, mu...
3	[ribuan, ulama, deklarasikan, dukung, ganjar, ...
4	[ribuan, warga, manado, sambut, ganjar, torang...

Pada Tabel 4.5 di atas menunjukkan hasil dari *tokenizing* yang mana akan memisahkan atau pemisahan dari suatu kalimat menjadi kata per kata atau bagian tertentu. Pada tahapan *tokenizing* terdapat nilai *frequency distribution tokenizing* pada hasil *tokenizing* yang telah dilakukan.

Listing source code untuk *frequency distribution* dapat dilihat Tabel 4.6 berikut ini:

Tabel 4.6. *Source Code Tokenizing*

No	Source Code
1	`def freqDist_wrapper(text):` `  return FreqDist(text)`
2	`TWEET_DATA['tweet_tokens_fdist'] =` `TWEET_DATA['tweet_tokens'].apply(freqDist_wrapper)`

3	`print('Frequency Tokens : \n')`
	`print(TWEET_DATA['tweet_tokens_fdist'].head().`
	`Apply(lambda x : x.most_common())`

Keterangan kode program *Source Code Frequency Distribution* di atas:

a) *Cell 1*, memadukan proses distribusi frekuensi setiap kata.

b) *Cell 2*, menerapkan proses distribusi frekuensi setiap kata.

c) *Cell 3*, melihat atau menampilkan hasil dari *running* kode program.

Hasilnya dapat dilihat pada Tabel 4.7 berikut ini:

Tabel 4.7. *Frequency Distribution*

	Tweet
0	[(kemiskinan, 1), (jateng, 1), (hingga, 1), (c...
1	[(warga, 2), (dan, 2), (kenaikan, 2), (isa, 2)...
2	[(bakal, 1), (capres, 1), (ganjar, 1), (pranow...
3	[(ganjar, 2), (ribuan, 1), (ulama, 1), (deklar...
4	[(warga, 2), (manado, 2), (ribuan, 1), (sambut...

Pada Tabel 4.7 di atas menunjukkan hasil dari *frequency distribution tokenizing* yang mana untuk menggambarkan berapa kali suatu kata diulang dalam kalimat.

4.2.3. *Filtering (Stopwords Removal)*

Filtering (*Stopwords Removal*) pada *DataFrame tweet* di atas menggunakan *library* dari *nltk.corpus* untuk *stopwords*. *Listing source code* dari *filtering (stopwords removal)* dapat dilihat Tabel 4.8 berikut ini:

Tabel 4.8. *Source Code Filtering (Stopwords Removal)*

No	Source Code
1	`from nltk.corpus import stopwords` `nltk.download("stopwords")`
2	`list_stopwords = stopwords.words('indonesian')`
3	`list_stopwords.extend(["yg", "dg", "rt", "dgn", "ny", "d", 'klo', 'kalo', 'amp', 'biar', 'bikin', 'bilang', 'gak', 'ga', 'krn', 'nya', 'nih', 'sih', 'si', 'tau', 'tdk', 'tuh', 'utk', 'ya', 'jd', 'jgn', 'sdh', 'aja', 'n', 't', 'nyg', 'hehe', 'pen', 'u', 'nan', 'loh', 'rt', '&amp', 'yah'])`
4	`list_stopwords = set(list_stopwords)`
5	`def stopwords_removal(words):` `  return [word for word in words if word not in list_stopwords]`

Tabel 4.8. (Lanjutan)

6	`TWEET_DATA['tweet_tokens_WSW'] = TWEET_DATA['tweet_tokens'].apply(stopwords_removal)`
7	`print(TWEET_DATA['tweet_tokens_WSW'].head())`

Keterangan kode program *Source Code Case Folding* di atas:

a) *Cell 1*, pemasangan *library* dari *swifter* dan *PySastrawi*.

b) *Cell 2*, mendapatlan *stopword* indonesia.

c) *Cell 3*, menambahkan *stopword* tambahan.

d) *Cell 4*, konversi *list* ke kamus.

e) *Cell 5*, hapus *stopword* pada daftar token.

f) *Cell 6*, menerapkan proses *stopword* pada daftar token.

g) *Cell 7*, melihat atau menampilkan hasil dari *running* kode program.

Hasilnya dapat dilihat pada Tabel 4.9 berikut ini:

Tabel 4.9. *Filtering (Stopwords Removal)*

	Tweet
0	[kemiskinan, jateng, capres, boneka, elektabil...
1	[disambut, warga, tokoh, gereja, rayakan, kena...
2	[capres, ganjar, pranowo, nampaknya, gencar, l...
3	[ribuan, ulama, deklarasikan, dukung, ganjar, ...
4	[ribuan, warga, manado, sambut, ganjar, torang...

Pada Tabel 4.9 di atas menunjukkan hasil dari *filtering (stopwords removal)* yang mana untuk memisahkan atau pemilihan kata penting yang bersangkutan atau mewakili kalimat dari dataset.

4.2.4. *Stemming*

Stemming pada *DataFrame tweet* di atas menggunakan *library* dari *swifter* dan *PySastrawi*. *Listing source code* dari *stemming* dapat dilihat Tabel 4.10 berikut ini:

Tabel 4.10. *Source Code Stemming*

No	Source Code
1	`!pip install swifter` `!pip install PySastrawi`
2	`from Sastrawi.Stemmer.StemmerFactory import StemmerFactory` `import swifter`
3	`factory = StemmerFactory()` `stemmer = factory.create_stemmer()`

4	`def stemmed_wrapper(term):`
	`  return stemmer.stem(term)`
	`term_dict = {}`
	`for document in TWEET_DATA['tweet_tokens_WSW']:`
	`  for term in document:`
	`  if term not in term_dict:`
	`  term_dict[term] = ' '`
	`print(len(term_dict))`
	`print("------------------------")`
	`for term in term_dict:`
	`  term_dict[term] = stemmed_wrapper(term)`
	`  print(term,":" ,term_dict[term])`
	`print(term_dict)`
	`print("------------------------")`

5	`def get_stemmed_term(document):` `  return [term_dict[term] for term in document]`

Tabel 4.10. (Lanjutan)

6	`TWEET_DATA['tweet_tokens_stemmed'] = TWEET_DATA['tweet_tokens_WSW'].swifter.apply (get_stemmed_term)`
7	`print(TWEET_DATA['tweet_tokens_stemmed'])`

Keterangan kode program *Source Code Stemming* di atas:

a) *Cell 1*, mendapatkan *stopword* dari NLTK.

b) *Cell 2*, impor paket *library* Sastrawi.

c) *Cell 3*, membuat *stemmer*.

d) *Cell 4*, proses *stemming*.

e) *Cell 5*, memadukan proses *stemming* ke kerangka data.

f) *Cell 6*, menerapkan proses *stemming*.

g) *Cell 7*, melihat atau menampilkan hasil dari *running* kode program.

Hasil prosesnya dapat dilihat pada Gambar 4.1 berikut ini:

```
rayakan : raya                pedagang : dagang
kenaikan : naik               kaki : kaki
isa : isa                     kalteng : kalteng
almasih : almasih             dibutuhkan : butuh
pdi : pdi                     dana : dana
perjuangan : juang            logistik : logistik
menghadiri : hadir            berkompetisi : kompetisi
peringatan : ingat            nabil : nabil
tomohon : tomohon             tanah : tanah
sulawesi : sulawesi           pasundan : pasundan
utara : utara                 wakili : wakil
sambutan : sambut             menyematkan : semat
```

Gambar 4.1. Proses *Stemming*

Pada Gambar 4.1 di atas memperlihatkan hasil dari proses *stemming* dalam dataset yang mana setiap kata diubah diuraikan dan menjadi kata dasar. Hasilnya dapat dilihat pada Tabel 4.11 berikut ini:

Tabel 4.11. *Stemming*

	Tweet
0	[miskin, jateng, capres, boneka, elektabilitas...
1	[sambut, warga, tokoh, gereja, raya, naik, isa...
2	[capres, ganjar, pranowo, nampaknya, gencar, l...
3	[ribu, ulama, deklarasi, dukung, ganjar, prano...
4	[ribu, warga, manado, sambut, ganjar, torang, ...

Pada Tabel 4.11 di atas menunjukkan hasil dari *stemming* dalam dataset yang mana setiap kata telah diuraikan dan menjadi kata dasar.

4.3. *TF-IDF*

Pada tahap TF-IDF dalam penelitian ini juga melakukan serangkaian pembobotan dengan tahapan berupa *ranking* dan *vector space matrix*. *Listing source code* dari *TF-IDF* dapat dilihat Tabel 4.12 berikut ini:

Tabel 4.12. *Source Code TF-IDF*

No	Source Code

1	`def calc_TF(document):`
	`  TF_dict = {}`
	`  for term in document:`
	`  if term in TF_dict:`
	`  TF_dict[term] += 1`
	`  else:`
	`  TF_dict[term] = 1`

Tabel 4.12. (Lanjutan)

2	for term in TF_dict: TF_dict[term] = TF_dict[term] / len(document) return TF_dict TWEET_DATA2["TF_dict"] = TWEET_DATA2['tweet_list'].apply(calc_TF) TWEET_DATA2["TF_dict"].head()
3	index = 90 print('%20s' % "term", "\t", "TF\n") for key in TWEET_DATA2["TF_dict"][index]: print('%20s' % key, "\t", TWEET_DATA2["TF_dict"][index][key])

4	def calc_DF(tfDict):
	count_DF = {}
	for document in tfDict:
	for term in document:
	if term in count_DF:
	count_DF[term] += 1
	else:
	count_DF[term] = 1
	return count_DF
	DF = calc_DF(TWEET_DATA2["TF_dict"])
	print(DF)

5	n_document = len(TWEET_DATA2)
	def calc_IDF(__n_document, __DF):
	IDF_Dict = {}
	for term in __DF:
	IDF_Dict[term] = np.log(__n_document /
	(__DF[term] + 1))
	return IDF_Dict

6	IDF = calc_IDF(n_document, DF)

Tabel 4.12. (Lanjutan)

7	`def calc_TF_IDF(TF):`
	`  TF_IDF_Dict = {}`
	`  #For each word in the review, we multiply its TD and its IDF`
	`  for key in TF:`
	`  TF_IDF_Dict[key] = TF[key] * IDF[key]`
	`  return TF_IDF_Dict`

8	`TWEET_DATA2["TF-IDF_dict"] = TWEET_DATA2["TF_dict"].apply(calc_TF_IDF)`

9	`index = 90`
	`print('%20s' % "term", "\t", '%10s' % "TF", "\t", '%20s' % "TF-IDF\n")`
	`for key in TWEET_DATA2["TF-IDF_dict"][index]:`
	`  print('%20s' % key, "\t",`

10	`TWEET_DATA2["TF_dict"][index][key] ,"\t" ,` `TWEET_DATA2["TF-IDF_dict"][index][key])`

Keterangan kode program *Source Code TF-IDF* di atas:

a) *Cell 1*, menghitung berapa kali kata tersebut muncul dalam ulasan.

b) *Cell 2*, menghitung *TF* untuk setiap kata.

c) *Cell 3*, memeriksa hasil *TF*.

d) *Cell 4*, malankan melalui kamus *TF* setiap dokumen dan *increment count Dict's* (*term, doc*).

e) *Cell 5*, menerapkan proses dari *increment count Dict's* (*term, doc*).

f) *Cell 6*, menyimpan kamus *IDF*.

g) *Cell 7*, mengalikan *TD* dan *IDF* untuk setiap kata dalam ulasan.

h) *Cell 8*, menyimpan seri *TF-IDF*.

i) *Cell 9*, memeriksa hasil *TF-IDF*.

j) *Cell 7*, melihat atau menampilkan hasil dari *running* kode program.

Hasilnya dapat dilihat pada Tabel 4.13 berikut ini:

Tabel 4.13. *TF-IDF*

Term	TF	TF-IDF
jalan	0.08333333333333333	0.3758216671819805
pikir	0.08333333333333333	0.5418575142394977
jokowi	0.08333333333333333	0.1922998029014658
ganjar	0.08333333333333333	0.00658693394503774
pranowo	0.08333333333333333	0.009711151354662634
manut	0.08333333333333333	0.5418575142394977
wae	0.08333333333333333	0.5418575142394977
capres	0.08333333333333333	0.011893025183466271
capresan	0.08333333333333333	0.4712493592072307
pdip	0.08333333333333333	0.16178502031520509
boneka	0.08333333333333333	0.44152644721233636

Pada Tabel 4.13 di atas menunjukkan hasil dari *TF-IDF* dalam pembobotan dari masing-masing kata dasar.

Listing source code dari *TF-IDF Rank* dapat dilihat Tabel 4.14 berikut ini:

Tabel 4.14. *Source Code TF-IDF Rank*

No	Source Code
1	`TF_IDF_Vec_List = np.array(TWEET_DATA2["TF_IDF_Vec"].to_list())`

2	`sums = TF_IDF_Vec_List.sum(axis=0)`
	`data = []`
	`for col, term in enumerate(unique_term):`
	`  data.append((term, sums[col]))`
	`ranking = pd.DataFrame(data, columns=['term', 'rank'])`
	`ranking.sort_values('rank', ascending=False)`

Keterangan kode program *Source Code TF-IDF* di atas:

a) *Cell 1*, mengkonversi *series* ke *list*.

b) *Cell 2*, mumlahkan vektor elemen dalam *axis=0*.

Hasilnya dapat dilihat pada Tabel 4.15 berikut ini:

Tabel 4.15. *TF-IDF Rank*

	Term	Rank
3	dukung	102.722318
5	prabowo	56.221091
18	ri	55.421328
4	presiden	50.356614
9	subianto	47.524242
8	elektabilitas	47.110847
11	jokowi	44.772443
6	pdip	44.297433
7	survei	44.038691
31	mahasiswa	41.199808
17	masyarakat	41.164268

Pada Tabel 4.15 di atas menunjukkan hasil dari *ranking* dari *TF*-IDF dari masing-masing kata dasar.

Listing source code dari *Vector Space Matrix* dapat dilihat Tabel 4.16 berikut ini:

Tabel 4.16. *Source Code Vector Space Matrix*

No	Source Code
1	`from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer` `from sklearn.preprocessing import normalize`

Tabel 4.16. (Lanjutan)

2	max_features = 100 cvect = CountVectorizer(max_features=max_features) TF_vector = cvect.fit_transform(TWEET_DATA2["tweet_join"])
3	normalized_TF_vector = normalize(TF_vector, norm='l1', axis=1)
4	tfidf = TfidfVectorizer(max_features=max_features, smooth_idf=False) tfs = tfidf.fit_transform(TWEET_DATA2["tweet_join"]) IDF_vector = tfidf.idf_
5	tfidf_mat = TF_vector.multiply(IDF_vector).toarray()
6	cvect = CountVectorizer(max_features=max_features, ngram_range=(1,3)) counts = cvect.fit_transform(TWEET_DATA2["tweet_join"])
7	normalized_counts = normalize(counts, norm='l1', axis=1) tfidf = TfidfVectorizer(max_features=max_features, ngram_range=(1,3), smooth_idf=False) tfs = tfidf.fit_transform(TWEET_DATA2["tweet_join"])
8	tfidf_mat = normalized_counts.multiply(IDF_vector).toarray() dfbtf = pd.DataFrame(data=tfidf_mat,columns=[a]) dfbtf.head(100)

Keterangan kode program *Source Code Vector Space Matrix* di atas:

a) *Cell 1*, impor *library* dari *nltk*.

b) *Cell 2*, kalkulasi *TF vector*.

c) *Cell 3*, normalisasi *TF vector*.

d) *Cell 4*, kalkulasi *IDF*.

e) *Cell 5*, hitung *TF x DF* sehingga dihasilkan *TF-IDF matrix / vector*.

f) *Cell 6*, *ngram-range* (1,3) untuk menggunakan *unigram, bigram, trigram*.

g) *Cell 7*, normalisasi perhitungan *TF-IDF*.

h) *Cell 8*, menampilkan hasil kalkulasi *TF-IDF*.

Hasilnya dapat dilihat pada Tabel 4.17 berikut ini:

Tabel 4.17. *Vector Space Matrix*

	calon	sulawesi utara	warga	warga manado
0	0.000000	0.000000	0.000000	0.000000
1	0.000000	0.350859	0.595727	0.000000
2	0.000000	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000
4	0.347507	0.350859	0.695014	0.845757

Pada Tabel 4.17 di atas menunjukkan hasil dari *Vector Space Matrix* dari masing-masing himpunan objek.

4.4. *Labeling* Sentimen

Penelitian ini menggunakan pelabelan dengan otomatis dengan *library vader lexicon* untuk melakukan pendekatan teknologi sentimen dengan proses perhitungan baku dalam kata dan kalimat, juga untuk menghindari *human error* dengan mengubahnya terlebih dahulu *text* dataset *tweet* yang telah melalui tahapan *preprocessing* berbahasa indonesia baku menjadi *text* bahasa inggris yang valid dengan maksud dari opini pada dataset terkait. *Listing source code* dari *Labeling* Sentimen dapat dilihat Tabel 4.18 berikut ini:

Tabel 4.18. *Source Code Labeling* Sentimen *Vader Lexicon*

No	Source Code

1	`import pandas as pd`
	`import numpy as np`
	`import nltk`
	`from nltk.sentiment.vader import SentimentIntensityAnalyzer`
	`nltk.download("vader_lexicon")`

2	`sentiments = SentimentIntensityAnalyzer()`
3	`TWEET_DATA3["Positive"] = [sentiments.polarity_scores(i)["pos"] for naive in TWEET_DATA3["Tweet_en"]]`
4	`TWEET_DATA3["Neugative"] = [sentiments.polarity_scores(i)["neg"] for naive in TWEET_DATA3["Tweet_en"]]`

Tabel 4.18. (Lanjutan)

5	`TWEET_DATA3["Neutral"] = [sentiments.polarity_scores(i)["neu"] for naive in TWEET_DATA3["Tweet_en"]]`
6	`TWEET_DATA3["Compound"] = [sentiments.polarity_scores(i)["compound"] for naive in TWEET_DATA3["Tweet_en"]]`
7	`TWEET_DATA3.head()`

7	`score = TWEET_DATA3["Compound"].values`
	`sentiment = []`
	`for naive in score:`
	`  if naive >= 0.05 :`
	`  sentiment.append('Positif')`
	`  elif naive <= -0.05 :`
	`  sentiment.append('Negatif')`
	`  else:`
	`  sentiment.append('Netral')`
	`TWEET_DATA3["Sentiment"] = sentiment`
	`TWEET_DATA3.head()`

Keterangan kode program *Source Code Labeling* Sentimen di atas:

a) *Cell 1*, impor *library* dari *padas, numpy,* dan *nltk.*

b) *Cell 2*, inisialisasi pendefinisian program *SentimentIntensityAnalyzer* dari *nltk.*

c) *Cell 3*, pembobotan atau pemberian *score* sentimen positif.

d) *Cell 4*, pembobotan atau pemberian *score* sentimen negatif.

e) *Cell 5*, pembobotan atau pemberian *score* sentimen netral.

f) *Cell 6,* menggabungkan hasil pembobotan atau pemberian *score* sentimen.

g) *Cell 7*, menampilkan hasil kode program di atas.

h) *Cell 8*, menampilkan hasil *labeling* sentimen positif, negatif, dan netral.

Hasilnya dapat dilihat pada Tabel 4.19 berikut ini:

Tabel 4.19. Hasil Perhitungan *Labeling* Sentimen *Vader Lexicon*

Tweet	Tweet_en	Positive	Neugative	Neutral	Compound	Sentiment
kemiskinan jateng hingga capres boneka bikin e...	Central Java poverty to doll presidential cand...	0.000	0.231	0.769	-0.5106	Negatif
disambut warga dan tokoh gereja yang tengah ra...	Welcomed by residents and church leaders who a...	0.255	0.000	0.745	0.9136	Positif
bakal capres ganjar pranowo nampaknya mulai ge...	Ganjar Pranowo Candidates seem to start aggres...	0.058	0.102	0.841	-0.2500	Negatif
ribuan ulama deklarasikan dukung ganjar pranow...	Thousands of scholars declare supporting Ganja...	0.301	0.000	0.699	0.6808	Positif
ribuan warga manado sambut ganjar torang pe pr...	Thousands of Manado residents welcomed Ganjar ...	0.128	0.123	0.749	0.0258	Netral

Pada Tabel 4.19 di atas menunjukkan hasil dari perhitungan *labeling* sentimen dari penggunaan *library vader lexicon*. Setiap *text* yang menjadi suatu kalimat yang telah melalui proses *preprocessing* akan terlebih dahulu di-*translate* dari bahasa indonesia ke bahasa inggris untuk penyesuaian dengan standar *library* yang hanya membaca kalimat dalam bahasa inggris. Sentimen positif, negatif, dan netral mengacu pada penjumlahan terhadap nilai *compound* yang telah diakumulasikan. *Labeling* secara otomatis memiliki kelebihan untuk mengefektifkan waktu dan menghindari *human error* dalam melakukan pelabelan. Namun, kekurangan dari *labeling* otomatis tidak dapat direkomendasikan jika dataset penelitian yang digunakan merupakan dari bidang kedokteran yang

memungkinkan akan mengakibatkan pembiasan makna dan istilah yang begitu kompleks dalam mengklasifikasikan *text*.

4.5. Persentase *Sentiment Analysis*

Pada tahap ini merupakan persentase sentimen dari masing-masing *tweet* dari dataset. Peneliti menggunakan *library* dari *matplotlib, seaborn,* dan *numpy*. *Listing source code* persentase *sentiment analysis* dapat dilihat Tabel 4.20 berikut ini:

Tabel 4.20. *Source Code* Persentase *Sentiment Analysis*

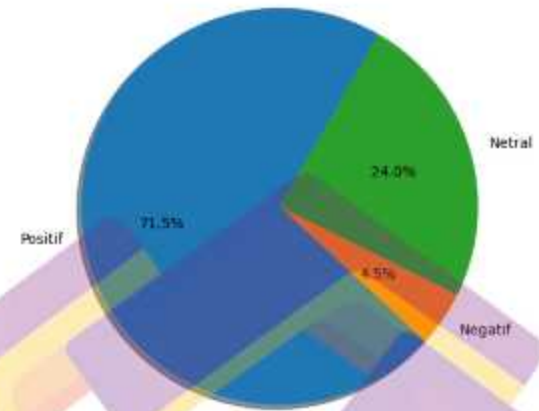
No	Source Code

1	`TWEET_DATA3.tail()`
	`TWEET_DATA3_filtered =`
	`TWEET_DATA3[TWEET_DATA3['Sentiment'] ==`
	`"Positif"]`
	`TWEET_DATA3_filtered =`
	`TWEET_DATA3[TWEET_DATA3['Sentiment'] ==`
	`"Negatif"]`
	`TWEET_DATA3_filtered =`
	`TWEET_DATA3[TWEET_DATA3['Sentiment'] == "Netral"]`

2	`Persentase =`
	`[np.count_nonzero(TWEET_DATA3['Sentiment'] ==`
	`'Positif'),`
	`np.count_nonzero(TWEET_DATA3['Sentiment'] ==`
	`'Negatif'),`
	`np.count_nonzero(TWEET_DATA3['Sentiment'] ==`
	`'Netral')]`
	`label = ['Positif','Negatif', 'Netral']`

3	`plt.pie(Persentase, labels = label, radius=1.3,`
	`startangle=60, autopct='%.1f%%', shadow=True)`
	`plt.show()`

Hasilnya dapat dilihat pada Gambar 4.2 berikut ini:



Gambar 4.2. Persentase *Sentiment Analysis*

Pada Gambar 4.2 di atas menunjukkan hasil dari persentase *sentiment analysis* yang mana sentimen positif masih mendominasi sebesar 71.5%, sentimen netral sebesar 24.0%, dan sentimen negatif sebesar 4.5%.

Opini yang tersaji pada data terkait merupakan dari kalangan kritis Pegiat Media Sosial terhadap politik daerah Provinsi Jawa Tengah, Aktivis Oposisi, dan hasil survei dari Korporat Media terkemuka. Sentimen negatif, sebagian besar tersaji atas opini dari kalangan oposisi maupun pegiat media sosial yang kritis terhadap politik dan kondisi perkembangan daerah Provinsi Jawa Tengah. Sehingga, pendekatan maupun gerakan kooperatif tertentu dapat dilakukan untuk memengaruhi nilai elektabilitas.

4.6. Pengujian Algoritma

Penelitian ini menggunakan dua pemodelan pengujian algoritma *K-Nearest Neighbors* (KNN) dan *Naïve Bayes* sebagai komparasi tingkat akurasi dari hasil masing-masing pengujian.

4.6.1. *K-Nearest Neighbors* (KNN)

Pada tahap pengujian ini, pembagian dataset untuk *splitting data* antara 80% data *training* dan 20% *data testing*, dan peneliti menggunakan *library* dari *sklearn.neighbors* untuk *KneighborsClassifier*, *sklearn.metrics* untuk *accuracy_score, precision_score, recall_score*, dan *f1_score, sklearn.metrics untuk classification_report* dan *confusion_matrix*. Pengujian ini akan melakukan beberapa percobaan pada nilai *k* dimulai dari 3, 5, 7, 9, 11, 13, dan 15 sebagai perbandingan variabel jumlah tetangga terdekat yang akan diambil untuk proses klasifikasi demi mendapatkan perbandingan yang signifikan.

Listing source code dari *K-Nearest Neighbors* (KNN) nilai $k = 3$ dapat dilihat Tabel 4.21 berikut ini:

Tabel 4.21. *Source Code K-Nearest Neighbors* (KNN) nilai $k = 3$

No	Source Code

1	`from sklearn.neighbors import KneighborsClassifier`
	`from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score`
	`from sklearn.metrics import classification_report`

Tabel 4.21. (Lanjutan)

2	clf = KneighborsClassifier(n_neighbors=3, weights= 'distance').fit(x_train, y_train) predicted = clf.predict(x_test)
3	print("KNN Accuracy:" , accuracy_score(y_test,predicted))
4	print("KNN Precision:" , precision_score(y_test,predicted, average="binary", pos_label="Negatif"))
5	print("KNN Recall:" , recall_score(y_test,predicted, average="binary", pos_label="Negatif"))
6	print("KNN F1_Score:" , f1_score(y_test,predicted, average="binary", pos_label="Negatif"))
7	print(f'Confusion Matrix:\n {confusion_matrix(y_test, predicted)}') print('========== ==========\n') print(classification_report(y_test, predicted, zero_division=0))
8	matrix = confusion_matrix(y_test, predicted) matrix predictions = clf.predict(x_test) cm = confusion_matrix(y_test, predictions, labels=clf.classes_) disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=clf.classes_) disp.plot() plt.show()

Keterangan kode program *Source Code K-Nearest Neighbors* (KNN) nilai $k = 3$ di atas:

a) *Cell 1*, impor *library* dari *sklearn*.

b) *Cell 2*, inisialisasi atau pendefinisian program *SentimentIntensityAnalyzer* dari *nltk* untuk *K-Nearest Neighbors* dengan nilai $k = 3$.

c) *Cell 3*, menampilkan hasil *Accuracy* dari pengujian.

d) *Cell 4*, menampilkan hasil *Precision* dari pengujian.

e) *Cell 5*, menampilkan hasil *Recall* dari pengujian.

f) *Cell 6*, menampilkan hasil *F1-Score* dari pengujian.

g) *Cell 7*, menampilkan hasil *Confusion Matrix* dari pengujian.

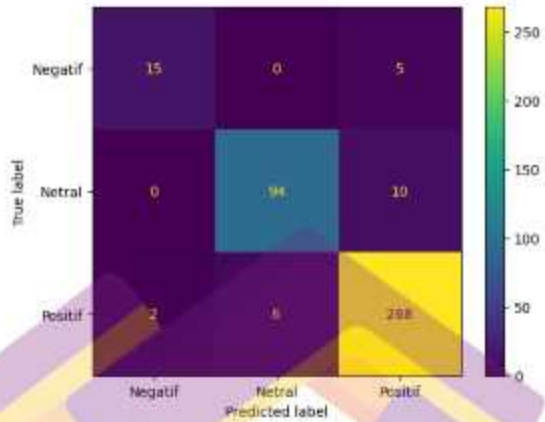
h) *Cell 8*, menampilkan visualisasi *Confusion Matrix* dari pengujian.

Hasil dari pengujian *K-Nearest Neighbors* (KNN) dengan nilai $k = 3$ dapat dilihat pada Tabel 4.22 berikut ini:

Tabel 4.22. Hasil Pengujian *K-Nearest Neighbors* (KNN) nilai $k = 3$

Accuracy	0.9425	94.25%
Precision	0.923116469202522	92.31%
Recall	0.8749535488665923	87.49%
F1-Score	0.8970781788839245	89.70%

Berdasarkan tabel di atas, hasil dari pengujian *K-Nearest Neighbors* (KNN) nilai $k = 3$ memperoleh nilai *accuracy* sebesar 94.25%, *precision* sebesar 91.31%, *recall* sebesar 87.49%, dan *f1-score* sebesar 89.70%. Hasil *confusion matrix* dari pengujian *K-Nearest Neighbors* (KNN) dapat dilihat Gambar 4.3 bawah ini:



Gambar 4.3. *Confusion Matrix K-Nearest Neighbors* (KNN) nilai $k = 3$

Berdasarkan gambar visualisasi *plot confusion* matrix di atas kita dapat melihat bahwa prediksi yang paling akurat yaitu pada label Positif. Sedangkan yang terburuk merupakan pada label Negatif karena prediksi yang salah lebih sedikit dari prediksi yang benar. Pada pengujian ini, nilai akurasi adalah 94.25%.

Listing source code dari *K-Nearest Neighbors* (KNN) nilai $k = 5$ dapat dilihat Tabel 4.23 berikut ini:

Tabel 4.23. *Source Code K-Nearest Neighbors* (KNN) nilai $k = 5$

No	Source Code

1	`from sklearn.neighbors import KneighborsClassifier`
	`from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score`
	`from sklearn.metrics import classification_report`

Tabel 4.23. (Lanjutan)

2	clf = KneighborsClassifier(n_neighbors=5, weights= 'distance').fit(x_train, y_train) predicted = clf.predict(x_test)
3	print("KNN Accuracy:" , accuracy_score(y_test,predicted))
4	print("KNN Precision:" , precision_score(y_test,predicted, average="binary", pos_label="Negatif"))
5	print("KNN Recall:" , recall_score(y_test,predicted, average="binary", pos_label="Negatif"))
6	print("KNN F1_Score:" , f1_score(y_test,predicted, average="binary", pos_label="Negatif"))
7	print(f'Confusion Matrix:\n {confusion_matrix(y_test, predicted)}') print('========== ==========\n') print(classification_report(y_test, predicted, zero_division=0))
8	matrix = confusion_matrix(y_test, predicted) matrix predictions = clf.predict(x_test) cm = confusion_matrix(y_test, predictions, labels=clf.classes_) disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=clf.classes_) disp.plot() plt.show()

Keterangan kode program *Source Code K-Nearest Neighbors* (KNN) nilai $k = 5$ di atas:

a) *Cell 1*, impor *library* dari *sklearn*.

b) *Cell 2*, inisialisasi atau pendefinisian program *SentimentIntensityAnalyzer* dari *nltk* untuk *K-Nearest Neighbors* dengan nilai $k = 5$.

c) *Cell 3*, menampilkan hasil *Accuracy* dari pengujian.

d) *Cell 4*, menampilkan hasil *Precision* dari pengujian.

e) *Cell 5*, menampilkan hasil *Recall* dari pengujian.

f) *Cell 6*, menampilkan hasil *F1-Score* dari pengujian.

g) *Cell 7*, menampilkan hasil *Confusion Matrix* dari pengujian.

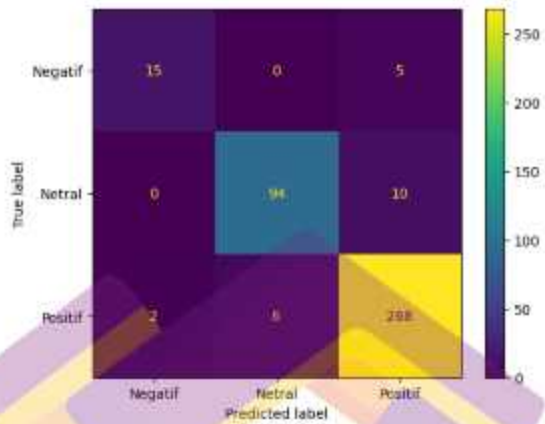
h) *Cell 8*, menampilkan visualisasi *Confusion Matrix* dari pengujian.

Hasil dari pengujian *K-Nearest Neighbors* (KNN) dengan nilai $k = 5$ dapat dilihat pada Tabel 4.24 berikut ini:

Tabel 4.24. Hasil Pengujian *K-Nearest Neighbors* (KNN) nilai $k = 5$

Accuracy	0.945	94.05%
Precision	0.9263436630876564	92.63%
Recall	0.8761612783351914	87.61%
F1-Score	0.8992111569697777	89.92%

Berdasarkan tabel di atas, hasil dari pengujian *K-Nearest Neighbors* (KNN) nilai $k = 5$ memperoleh nilai *accuracy* sebesar 94.05%, *precision* sebesar 92.63%, *recall* sebesar 87.61%, dan *f1-score* sebesar 89.92%. Hasil *confusion matrix* dari pengujian *K-Nearest Neighbors* (KNN) dapat dilihat Gambar 4.4 bawah ini:



Gambar 4.4. *Confusion Matrix K-Nearest Neighbors* (KNN) nilai $k = 5$

Berdasarkan gambar visualisasi *plot confusion* matrix di atas kita dapat melihat bahwa prediksi yang paling akurat yaitu pada label Positif. Sedangkan yang terburuk merupakan pada label Negatif karena prediksi yang salah lebih sedikit dari prediksi yang benar. Pada pengujian ini, nilai akurasi adalah 94.05%.

Listing source code dari *K-Nearest Neighbors* (KNN) nilai $k = 7$ dapat dilihat Tabel 4.25 berikut ini:

Tabel 4.25. *Source Code K-Nearest Neighbors* (KNN) nilai $k = 7$

No	Source Code

1	`from sklearn.neighbors import KneighborsClassifier`
	`from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score`
	`from sklearn.metrics import classification_report`

Tabel 4.25. (Lanjutan)

2	clf = KneighborsClassifier(n_neighbors=7, weights= 'distance').fit(x_train, y_train) predicted = clf.predict(x_test)
3	print("KNN Accuracy:" , accuracy_score(y_test,predicted))
4	print("KNN Precision:" , precision_score(y_test,predicted, average="binary", pos_label="Negatif"))
5	print("KNN Recall:" , recall_score(y_test,predicted, average="binary", pos_label="Negatif"))
6	print("KNN F1_Score:" , f1_score(y_test,predicted, average="binary", pos_label="Negatif"))
7	print(f'Confusion Matrix:\n {confusion_matrix(y_test, predicted)}') print('========== ==========\n') print(classification_report(y_test, predicted, zero_division=0))
8	matrix = confusion_matrix(y_test, predicted) matrix predictions = clf.predict(x_test) cm = confusion_matrix(y_test, predictions, labels=clf.classes_) disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=clf.classes_) disp.plot() plt.show()

Keterangan kode program *Source Code K-Nearest Neighbors* (KNN) nilai $k = 7$ di atas:

a) *Cell 1*, impor *library* dari *sklearn*.

b) *Cell 2*, inisialisasi atau pendefinisian program *SentimentIntensityAnalyzer* dari *nltk* untuk *K-Nearest Neighbors* dengan nilai $k = 7$.

c) *Cell 3*, menampilkan hasil *Accuracy* dari pengujian.

d) *Cell 4*, menampilkan hasil *Precision* dari pengujian.

e) *Cell 5*, menampilkan hasil *Recall* dari pengujian.

f) *Cell 6*, menampilkan hasil *F1-Score* dari pengujian.

g) *Cell 7*, menampilkan hasil *Confusion Matrix* dari pengujian.

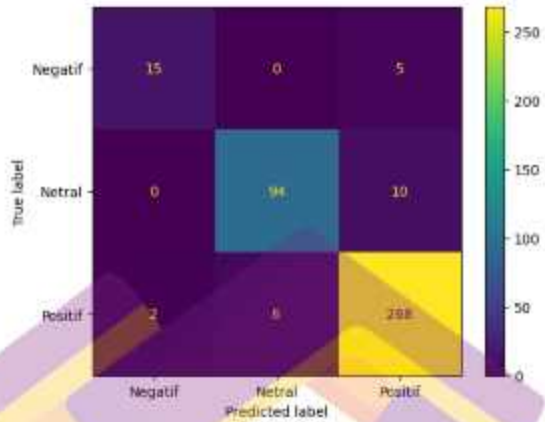
h) *Cell 8*, menampilkan visualisasi *Confusion Matrix* dari pengujian.

Hasil dari pengujian *K-Nearest Neighbors* (KNN) dengan nilai $k = 7$ dapat dilihat pada Tabel 4.26 berikut ini:

Tabel 4.26. Hasil Pengujian *K-Nearest Neighbors* (KNN) nilai $k = 7$

Accuracy	0.9325	93.25%
Precision	0.9489473684210527	94.89%
Recall	0.8641304347826088	86.41%
F1-Score	0.9003904592139887	90.03%

Berdasarkan tabel di atas, hasil dari pengujian *K-Nearest Neighbors* (KNN) nilai $k = 7$ memperoleh nilai *accuracy* sebesar 94.25%, *precision* sebesar 94.89%, *recall* sebesar 87.41%, dan *f1-score* sebesar 90.03%. Hasil *confusion matrix* dari pengujian *K-Nearest Neighbors* (KNN) dapat dilihat Gambar 4.5 bawah ini:



Gambar 4.5. *Confusion Matrix K-Nearest Neighbors* (KNN) nilai $k = 7$

Berdasarkan gambar visualisasi *plot confusion* matrix di atas kita dapat melihat bahwa prediksi yang paling akurat yaitu pada label Positif. Sedangkan yang terburuk merupakan pada label Negatif karena prediksi yang salah lebih sedikit dari prediksi yang benar. Pada pengujian ini, nilai akurasi adalah 94.25%.

Listing source code dari *K-Nearest Neighbors* (KNN) nilai $k = 9$ dapat dilihat Tabel 4.27 berikut ini:

Tabel 4.27. *Source Code K-Nearest Neighbors* (KNN) nilai $k = 9$

No	Source Code

1	`from sklearn.neighbors import KneighborsClassifier`
	`from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score`
	`from sklearn.metrics import classification_report`

Tabel 4.27. (Lanjutan)

2	`clf = KneighborsClassifier(n_neighbors=9, weights= 'distance').fit(x_train, y_train)` `predicted = clf.predict(x_test)`
3	`print("KNN Accuracy:" , accuracy_score(y_test,predicted))`
4	`print("KNN Precision:" , precision_score(y_test,predicted, average="binary", pos_label="Negatif"))`
5	`print("KNN Recall:" , recall_score(y_test,predicted, average="binary", pos_label="Negatif"))`
6	`print("KNN F1_Score:" , f1_score(y_test,predicted, average="binary", pos_label="Negatif"))`

7	`print(f'Confusion Matrix:\n {confusion_matrix(y_test, predicted)}')`
	`print('==========\n')`
	`print(classification_report(y_test, predicted, zero_division=0))`

8	`matrix = confusion_matrix(y_test, predicted)`
	`matrix`
	`predictions = clf.predict(x_test)`
	`cm = confusion_matrix(y_test, predictions, labels=clf.classes_)`
	`disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=clf.classes_)`
	`disp.plot()`
	`plt.show()`

Keterangan kode program *Source Code K-Nearest Neighbors* (KNN) nilai $k = 9$ di atas:

a) *Cell 1*, impor *library* dari *sklearn*.

b) *Cell 2*, inisialisasi atau pendefinisian program *SentimentIntensityAnalyzer* dari *nltk* untuk *K-Nearest Neighbors* dengan nilai $k = 9$.

c) *Cell 3*, menampilkan hasil *Accuracy* dari pengujian.

d) *Cell 4*, menampilkan hasil *Precision* dari pengujian.

e) *Cell 5*, menampilkan hasil *Recall* dari pengujian.

f) *Cell 6*, menampilkan hasil *F1-Score* dari pengujian.

g) *Cell 7*, menampilkan hasil *Confusion Matrix* dari pengujian.

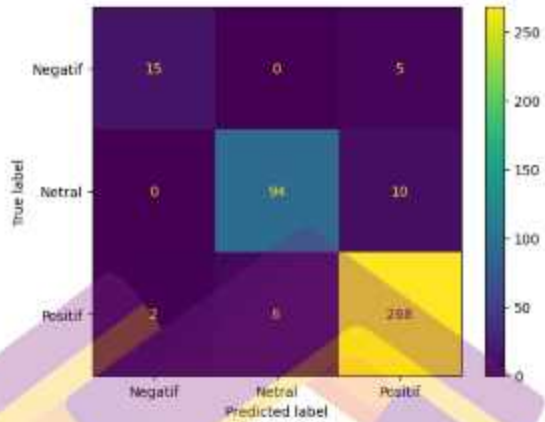
h) *Cell 8*, menampilkan visualisasi *Confusion Matrix* dari pengujian.

Hasil dari pengujian *K-Nearest Neighbors* (KNN) dengan nilai $k = 9$ dapat dilihat pada Tabel 4.28 berikut ini:

Tabel 4.28. Hasil Pengujian *K-Nearest Neighbors* (KNN) nilai $k = 9$

Accuracy	0.9375	93.75%
Precision	0.9537143188003011	95.37%
Recall	0.8550817539947975	85.50%
F1-Score	0.8954277350260135	89.54%

Berdasarkan tabel di atas, hasil dari pengujian *K-Nearest Neighbors* (KNN) nilai $k = 9$ memperoleh nilai *accuracy* sebesar 93.75%, *precision* sebesar 95.37%, *recall* sebesar 85.50%, dan *f1-score* sebesar 89.54%. Hasil *confusion matrix* dari pengujian *K-Nearest Neighbors* (KNN) dapat dilihat Gambar 4.6 bawah ini:



Gambar 4.6. *Confusion Matrix K-Nearest Neighbors* (KNN) nilai $k = 9$

Berdasarkan gambar visualisasi *plot confusion* matrix di atas kita dapat melihat bahwa prediksi yang paling akurat yaitu pada label Positif. Sedangkan yang terburuk merupakan pada label Negatif karena prediksi yang salah lebih sedikit dari prediksi yang benar. Pada pengujian ini, nilai akurasi adalah 93.75%.

Listing source code dari *K-Nearest Neighbors* (KNN) nilai $k = 11$ dapat dilihat Tabel 4.29 berikut ini:

Tabel 4.29. *Source Code K-Nearest Neighbors* (KNN) nilai $k = 11$

No	Source Code

1	`from sklearn.neighbors import KneighborsClassifier`
	`from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score`
	`from sklearn.metrics import classification_report`

Tabel 4.29. (Lanjutan)

2	clf = KneighborsClassifier(n_neighbors=11, weights= 'distance').fit(x_train, y_train) predicted = clf.predict(x_test)
3	print("KNN Accuracy:" , accuracy_score(y_test,predicted))
4	print("KNN Precision:" , precision_score(y_test,predicted, average="binary", pos_label="Negatif"))
5	print("KNN Recall:" , recall_score(y_test,predicted, average="binary", pos_label="Negatif"))
6	print("KNN F1_Score:" , f1_score(y_test,predicted, average="binary", pos_label="Negatif"))
7	print(f'Confusion Matrix:\n {confusion_matrix(y_test, predicted)}') print('==========\n') print(classification_report(y_test, predicted, zero_division=0))
8	matrix = confusion_matrix(y_test, predicted) matrix predictions = clf.predict(x_test) cm = confusion_matrix(y_test, predictions, labels=clf.classes_) disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=clf.classes_) disp.plot() plt.show()

Keterangan kode program *Source Code K-Nearest Neighbors* (KNN) nilai $k = 11$ di atas:

a) *Cell 1*, impor *library* dari *sklearn*.

b) *Cell 2*, inisialisasi atau pendefinisian program *SentimentIntensityAnalyzer* dari *nltk* untuk *K-Nearest Neighbors* dengan nilai $k = 11$.

c) *Cell 3*, menampilkan hasil *Accuracy* dari pengujian.

d) *Cell 4*, menampilkan hasil *Precision* dari pengujian.

e) *Cell 5*, menampilkan hasil *Recall* dari pengujian.

f) *Cell 6*, menampilkan hasil *F1-Score* dari pengujian.

g) *Cell 7*, menampilkan hasil *Confusion Matrix* dari pengujian.

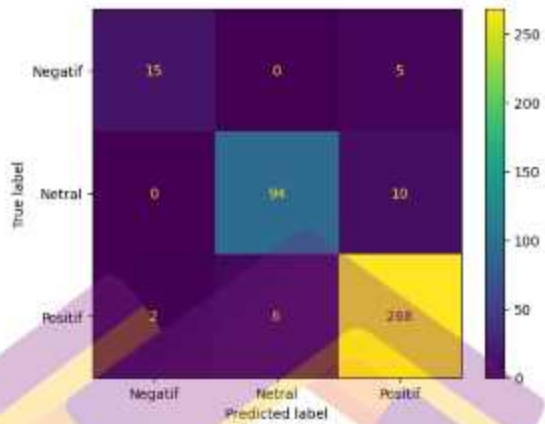
h) *Cell 8*, menampilkan visualisasi *Confusion Matrix* dari pengujian.

Hasil dari pengujian *K-Nearest Neighbors* (KNN) dengan nilai $k = 11$ dapat dilihat pada Tabel 4.30 berikut ini:

Tabel 4.30. Hasil Pengujian *K-Nearest Neighbors* (KNN) nilai $k = 11$

Accuracy	0.9325	93.25%
Precision	0.9528769664324189	95.28%
Recall	0.8466740988480118	84.66%
F1-Score	0.8904213964016462	89.90%

Berdasarkan tabel di atas, hasil dari pengujian *K-Nearest Neighbors* (KNN) nilai $k = 11$ memperoleh nilai *accuracy* sebesar 93.25%, *precision* sebesar 95.28%, *recall* sebesar 84.66%, dan *f1-score* sebesar 84.66%. Hasil *confusion matrix* dari pengujian *K-Nearest Neighbors* (KNN) dapat dilihat Gambar 4.7 bawah ini:



Gambar 4.7. *Confusion Matrix K-Nearest Neighbors* (KNN) nilai k = 11

Berdasarkan gambar visualisasi *plot confusion* matrix di atas kita dapat melihat bahwa prediksi yang paling akurat yaitu pada label Positif. Sedangkan yang terburuk merupakan pada label Negatif karena prediksi yang salah lebih sedikit dari prediksi yang benar. Pada pengujian ini, nilai akurasi adalah 93.25%.

Listing source code dari *K-Nearest Neighbors* (KNN) nilai k = 13 dapat dilihat Tabel 4.31 berikut ini:

Tabel 4.31. *Source Code K-Nearest Neighbors* (KNN) nilai k = 13

No	Source Code

1	`from sklearn.neighbors import KneighborsClassifier`
	`from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score`
	`from sklearn.metrics import classification_report`

Tabel 4.31. (Lanjutan)

2	clf = KneighborsClassifier(n_neighbors=13, weights= 'distance').fit(x_train, y_train) predicted = clf.predict(x_test)
3	print("KNN Accuracy:" , accuracy_score(y_test,predicted))
4	print("KNN Precision:" , precision_score(y_test,predicted, average="binary", pos_label="Negatif"))
5	print("KNN Recall:" , recall_score(y_test,predicted, average="binary", pos_label="Negatif"))
6	print("KNN F1_Score:" , f1_score(y_test,predicted, average="binary", pos_label="Negatif"))
7	print(f'Confusion Matrix:\n {confusion_matrix(y_test, predicted)}') print('========== ==========\n') print(classification_report(y_test, predicted, zero_division=0))
8	matrix = confusion_matrix(y_test, predicted) matrix predictions = clf.predict(x_test) cm = confusion_matrix(y_test, predictions, labels=clf.classes_) disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=clf.classes_) disp.plot() plt.show()

Keterangan kode program *Source Code K-Nearest Neighbors* (KNN) nilai k = 13 di atas:

a) *Cell 1*, impor *library* dari *sklearn*.

b) *Cell 2*, inisialisasi atau pendefinisian program *SentimentIntensityAnalyzer* dari *nltk* untuk *K-Nearest Neighbors* dengan nilai k = 13.

c) *Cell 3*, menampilkan hasil *Accuracy* dari pengujian.

d) *Cell 4*, menampilkan hasil *Precision* dari pengujian.

e) *Cell 5*, menampilkan hasil *Recall* dari pengujian.

f) *Cell 6*, menampilkan hasil *F1-Score* dari pengujian.

g) *Cell 7*, menampilkan hasil *Confusion Matrix* dari pengujian.

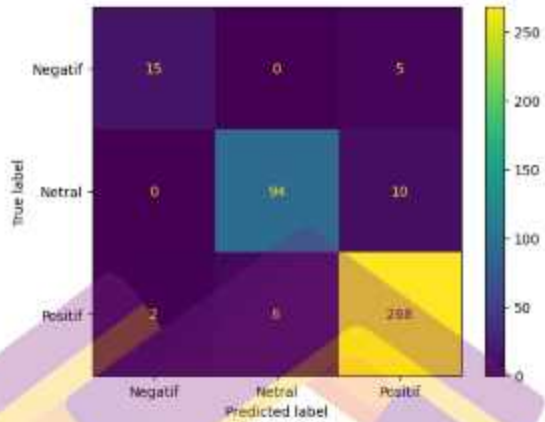
h) *Cell 8*, menampilkan visualisasi *Confusion Matrix* dari pengujian.

Hasil dari pengujian *K-Nearest Neighbors* (KNN) dengan nilai k = 13 dapat dilihat pada Tabel 4.32 berikut ini:

Tabel 4.32. Hasil Pengujian *K-Nearest Neighbors* (KNN) nilai k = 13

Accuracy	0.935	93.50%
Precision	0.9541997354497355	95.41%
Recall	0.84987922705314	84.98%
F1-Score	0.8928067399981551	89.28%

Berdasarkan tabel di atas, hasil dari pengujian *K-Nearest Neighbors* (KNN) nilai k = 13 memperoleh nilai *accuracy* sebesar 93.50%, *precision* sebesar 95.41%, *recall* sebesar 84.98%, dan *f1-score* sebesar 89.28%. Hasil *confusion matrix* dari pengujian *K-Nearest Neighbors* (KNN) dapat dilihat Gambar 4.8 bawah ini:



Gambar 4.8. *Confusion Matrix K-Nearest Neighbors* (KNN) nilai k = 13

Berdasarkan gambar visualisasi *plot confusion* matrix di atas kita dapat melihat bahwa prediksi yang paling akurat yaitu pada label Positif. Sedangkan yang terburuk merupakan pada label Negatif karena prediksi yang salah lebih sedikit dari prediksi yang benar. Pada pengujian ini, nilai akurasi adalah 93.50%.

Listing source code dari *K-Nearest Neighbors* (KNN) nilai k = 15 dapat dilihat Tabel 4.33 berikut ini:

Tabel 4.33. *Source Code K-Nearest Neighbors* (KNN) nilai k = 15

No	Source Code

1	`from sklearn.neighbors import KneighborsClassifier`
	`from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score`
	`from sklearn.metrics import classification_report`

Tabel 4.33. (Lanjutan)

2	clf = KneighborsClassifier(n_neighbors=15, weights= 'distance').fit(x_train, y_train) predicted = clf.predict(x_test)
3	print("KNN Accuracy:" , accuracy_score(y_test,predicted))
4	print("KNN Precision:" , precision_score(y_test,predicted, average="binary", pos_label="Negatif"))
5	print("KNN Recall:" , recall_score(y_test,predicted, average="binary", pos_label="Negatif"))
6	print("KNN F1_Score:" , f1_score(y_test,predicted, average="binary", pos_label="Negatif"))
7	print(f'Confusion Matrix:\n {confusion_matrix(y_test, predicted)}') print('========== ==========\n') print(classification_report(y_test, predicted, zero_division=0))
8	matrix = confusion_matrix(y_test, predicted) matrix predictions = clf.predict(x_test) cm = confusion_matrix(y_test, predictions, labels=clf.classes_) disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=clf.classes_) disp.plot() plt.show()

Keterangan kode program *Source Code K-Nearest Neighbors* (KNN) nilai k = 15 di atas:

a) *Cell 1*, impor *library* dari *sklearn*.

b) *Cell 2*, inisialisasi atau pendefinisian program *SentimentIntensityAnalyzer* dari *nltk* untuk *K-Nearest Neighbors* dengan nilai k = 15.

c) *Cell 3*, menampilkan hasil *Accuracy* dari pengujian.

d) *Cell 4*, menampilkan hasil *Precision* dari pengujian.

e) *Cell 5*, menampilkan hasil *Recall* dari pengujian.

f) *Cell 6*, menampilkan hasil *F1-Score* dari pengujian.

g) *Cell 7*, menampilkan hasil *Confusion Matrix* dari pengujian.

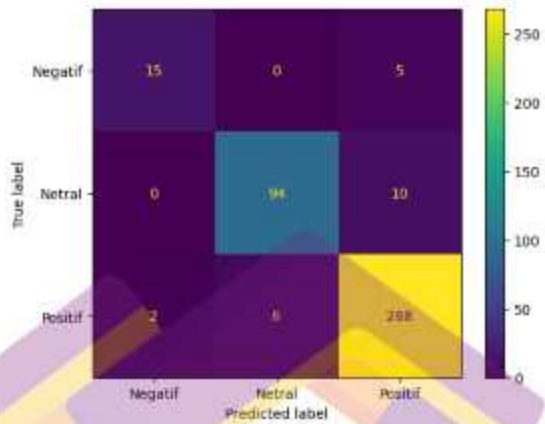
h) *Cell 8*, menampilkan visualisasi *Confusion Matrix* dari pengujian.

Hasil dari pengujian *K-Nearest Neighbors* (KNN) dengan nilai k = 15 dapat dilihat pada Tabel 4.34 berikut ini:

Tabel 4.34. Hasil Pengujian *K-Nearest Neighbors* (KNN) nilai k = 15

Accuracy	0.935	93.50%
Precision	0.9523543123543123	95.23%
Recall	0.8518766257896693	85.18%
F1-Score	0.8930756169609007	89.30%

Berdasarkan tabel di atas, hasil dari pengujian *K-Nearest Neighbors* (KNN) nilai k = 15 memperoleh nilai *accuracy* sebesar 93.50%, *precision* sebesar 95.23%, *recall* sebesar 85.18%, dan *f1-score* sebesar 89.30%. Hasil *confusion matrix* dari pengujian *K-Nearest Neighbors* (KNN) dapat dilihat Gambar 4.9 bawah ini:



Gambar 4.9. *Confusion Matrix K-Nearest Neighbors* (KNN) nilai k = 15

Berdasarkan gambar visualisasi *plot confusion* matrix di atas kita dapat melihat bahwa prediksi yang paling akurat yaitu pada label Positif. Sedangkan yang terburuk merupakan pada label Negatif karena prediksi yang salah lebih sedikit dari prediksi yang benar. Pada pengujian ini, nilai akurasi adalah 93.50%.

Perbandingan hasil pengujian dari *K-Nearest Neighbors* (KNN) dari masing-masing veriasi nilai *k* dapat dilihat pada Tabel 4.35 berikut ini:

Tabel 4.35. Perbandingan Hasil Pengujian *K-Nearest Neighbors* (KNN)

KNN	*Accuracy*	*Precision*	*Recall*	*F1-Score*
K = 3	94.25%	92.31%	87.49%	89.70%
K = 5	94.50%	92.63%	87.61%	89.92%
K = 7	93.25%	94.89%	86.41%	90.03%
K = 9	93.75%	95.37%	85.50%	89.54%
K = 11	93.25%	95.28%	84.66%	89.04%
K = 13	93.50%	95.41%	84.98%	89.28%
K = 15	93.50%	95.23%	85.18%	89.30%

Berdasarkan tabel di atas, hasil dari pengujian *K-Nearest Neighbors* (KNN) masing-masing nilai *k* memperoleh nilai *accuracy*, *precision*, *recall*, dan *f1-score* yang berbeda dan variatif. Hasil terbaik didapatkan dari pengujian nilai *k* = 5 yang mengacu pada tingkat akurasi pengujian yang mana nilai *accuracy* sebesar 94.50%, nilai *precision* sebesar 92.63%, nilai *recall* sebesar 87.61%, dan nilai *F1-Sore* sebesar 89.92%.

4.6.2. *Naïve Bayes*

Pada tahap pengujian ini, pembagian dataset untuk *splitting data* antara 80% data *training* dan 20% *data testing*, dan peneliti menggunakan *library* dari *sklearn.naive_bayes* untuk *MultinomialNB*, *sklearn.metrics* untuk *accuracy_score, precision_score, recall_score,* dan *f1_score*, *sklearn.metrics* untuk *classification_report* dan *confusion_matrix*. *Listing source code* dari *Naive Bayes* dapat dilihat Tabel 4.36 berikut ini:

Tabel 4.36. *Source Code Naïve Bayes*

No	Source Code

1	`from sklearn.naive_bayes import MultinomialNB`
	`from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score`
	`from sklearn.metrics import classification_report`

2	`clf = MultinomialNB().fit(x_train, y_train)` `predicted = clf.predict(x_test)`
3	`print("KNN Accuracy:" , accuracy_score(y_test,predicted))`
4	`print("KNN Precision:" , precision_score(y_test,predicted, average="binary", pos_label="Negatif"))`
5	`print("KNN Recall:" , recall_score(y_test,predicted, average="binary", pos_label="Negatif"))`
6	`print("KNN F1_Score:" , f1_score(y_test,predicted, average="binary", pos_label="Negatif"))`

Tabel 4.36. (Lanjutan)

7	`print(f'Confusion Matrix:\n {confusion_matrix(y_test, predicted)}')`
	`print('==========\n')`
	`print(classification_report(y_test, predicted, zero_division=0))`

8	`matrix = confusion_matrix(y_test, predicted)`
	`matrix`
	`predictions = clf.predict(x_test)`
	`cm = confusion_matrix(y_test, predictions, labels=clf.classes_)`
	`disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=clf.classes_)`
	`disp.plot()`
	`plt.show()`

Keterangan kode program *Source Code Naïve Bayes* di atas:

a) *Cell 1*, impor *library* dari *sklearn*.

b) *Cell 2*, inisialisasi atau pendefinisian program *SentimentIntensityAnalyzer* dari *nltk* untuk *Naïve Bayes*.

c) *Cell 3*, menampilkan hasil *Accuracy* dari pengujian.

d) *Cell 4*, menampilkan hasil *Precision* dari pengujian.

e) *Cell 5*, menampilkan hasil *Recall* dari pengujian.

f) *Cell 6*, menampilkan hasil *F1-Score* dari pengujian.

g) *Cell 7*, menampilkan hasil *Confusion Matrix* dari pengujian.

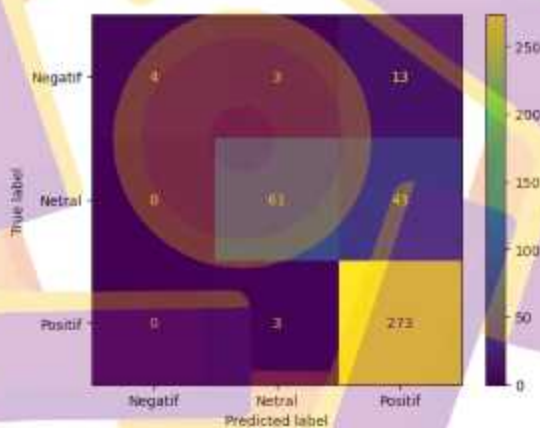
h) *Cell 8*, menampilkan visualisasi *Confusion Matrix* dari pengujian.

Hasil dari pengujian *Naïve Bayes* dapat dilihat pada Tabel 4.37 berikut ini:

Tabel 4.37. Hasil Pengujian *Naïve Bayes*

Accuracy	0.845	84.05%
Precision	0.913411665078861	91.34%
Recall	0.5918896321070234	59.18%
F1-Score	0.6497543215246564	64.97%

Berdasarkan tabel di atas, hasil dari pengujian *Naive Bayes* memperoleh nilai *accuracy* sebesar 84.05%, *precision* sebesar 91.34%, *recall* sebesar 59.18%, dan *f1-score* sebesar 64.97%. Hasil *confusion matrix* dari pengujian *Naïve Bayes* dapat dilihat pada Gambar 4.10 di bawah ini:



Gambar 4.10. *Confusion Matrix Naïve Bayes*

Berdasarkan gambar visualisasi *plot confusion* matrix di atas kita dapat melihat bahwa prediksi yang paling akurat yaitu pada label Positif. Sedangkan yang terburuk merupakan pada label Negatif karena prediksi yang salah lebih sedikit dari prediksi yang benar. Pada pengujian ini, nilai akurasi adalah 84.05%.

4.7. Analisis Perbandingan Penelitian

Sebagai bahan perbandingan dengan penelitian sebelumnya, berikut penulis sajikan analisis dari beberapa penelitian serupa dengan penggunaan pemodelan algoritma penelitian yang berbeda dapat dilihat pada Tabel 4.38 berikut ini:

Tabel 4.38. Analisis Perbandingan Penelitian

No	Author	Model Algoritma Penelitian	Akurasi
1.	(Bayhaqy dkk., t.t.)	*Decision Tree, K-Nearest Neighbors (K-NN), Naïve Bayes*	80%
2.	(Deviyanto dkk., 2018)	*K-Nearest Neighbors (K-NN)*	67,2%
3.	(Septian dkk., 2019)	*K-Nearest Neighbors (K-NN)*	79,99%
4.	(Salim & Mayary, 2020)	*Lexon Based, K-Neurest Neighbors (K-NN)*	86,91%
5.	(Duei Putri dkk., 2022)	*Naïve Bayes Classifier*	80%
6.	(Darwis dkk., t.t.)	*Naïve Bayes*	68.97%
7.	(Fajar dkk., t.t.)	*Naïve Bayes*	90%
8.	(Anjas Ramadhan & Budi Setiawan Ssi, t.t.)	*Naïve Bayes, Support Vector Machine*	88,57%
9.	(Rahayu dkk., 2022)	*K-Nearest Neighbors (K-NN)*	76.68%
10.	(Adhi Putra, 2021)	*K-Nearest Neighbors (K-NN)*	85,14%
11.	(Asro'i & Februariyanti, 2022)	*K-Nearest Neighbors (K-NN)*	69.5%
12.	(Yoga Syantara dkk., 2021)	*Naïve Bayes Classifier*	80.30%
13.	(Febriyani & Februariyanti, t.t.)	*Naïve Bayes Classifier*	57%
14.	(I Komang Andi Sugiarta dkk., 2023)	*Naïve Bayes, K-Nearest Neighbors (KNN)*	85,06%
15.	(Ndapamuri dkk., 2023)	*Support Vector Machine, K-Nearest Neighbors (K-NN), Naïve Bayes*	89,8%
16.	(Supriyanto dkk., t.t.)	*K-Nearest Neighbors (K-NN)*	84,93%
17.	(Permana & Sahara, 2023)	*K-Nearest Neighbors (K-NN)*	83.20%
18.	(Febriansyah dkk., 2023)	*K-Neasert Naighbors (K-NN)*	87,35%
19.	(MZ dkk., 2023)	*Naïve Bayes Classifier, K-Nearest Neighbors (K-NN)*	75.97%
20.	(Saputra & Pribadi, 2023)	*Naïve Bayes*	67%
21.	Penelitian Terbaru (Penulis)	*K-Nearest Neighbors (K-NN), Naïve Bayes*	94.50%

Berdasarkan tabel di atas, hasil dari penelitian terbaru menunjukkan akurasi lebih baik dengan nilai akurasi sebesar 94.50% dengan algoritma *K-Nearest Neighbors* (KNN) sebagai metode pengujian. Penulis melakukan perbaikan dan penyempurnaan dalam proses translasi bahasa dari dataset berhabasa indonesia yang seringkali terdapat satire dan majas maupun gaya sastra Bahasa Indonesia lainnya yang tidak ada pada Bahasa Inggris dan juga riskan terhadap kelas yang tidak *balance*.

BAB V

PENUTUP

5.1. Kesimpulan

Berdasarkan hasil penelitian di atas maka penulis dapat menyimpulkan bahwa:

a. Berdasarkan penelitian di atas dengan 80% data latih dan 20% data uji dapat disimpulkan bahwa akurasi dari masing-masing pemodelan dapat dipengaruhi oleh *splitting data* dan nilai *k* terkhusus untuk pemodelan *K-Nearest Neighbors*. Tingkat akurasi pada model *K-Nearest Neighbors* (KNN) nilai $k = 5$ memperoleh nilai *accuracy* sebesar 94.50%, nilai *precision* sebesar 92.63%, nilai *recall* sebesar 87.61%, dan nilai *F1-Score* sebesar 89.92%. Sedangkan pada model *Naïve Bayes* yang telah di implementasikan yang telah diujikan memperoleh nilai *accuracy* sebesar 84.05%, nilai *precision* sebesar 91.34%, nilai *recall* sebesar 59.18%, dan nilai *F1-Score* sebesar 64.97%. Sehingga dapat disimpulan bahwa pengujian dengan model *K-Nearest Neighbors* (KNN) pada penelitian ini memiliki akurasi terbaik dengan nilai *acuracy* sebesar 94.50% berada di atas dari model pengujian *Naïve Bayes* dengan nilai *accuracy* sebesar 84.05%.
b. Tahapan *preprocessing* sangat berpengaruh terhadap hasil akurasi ketika menggunakan *labeling* otomatis, khususnya pada bagian translasi bahasa dari Bahasa Indonesia yang penuh dengan satire dan majas maupun gaya satra lainnya yang tidak ada pada Bahasa Inggris.

c. Persentase *sentiment* dengan perbandingan 71.5% untuk sentimen positif, 24.0% untuk sentimen netral, dan 4.5% untuk sentimen negatif dapat menyimpulkan untuk memberikan informasi bahwa sebagian besar masyarakat masih mendominasi opini berupa sentimen positif terhadap Elektabilitas Ganjar Pranowo sebagai capres di tahun politik 2024.

5.2. Saran

Berdasarkan penelitian yang telah penulis lakukan, berikut beberapa saran untuk pengembangan penelitian ini, di antarannya adalah sebagai berikut:

a. Pada penelitian ini, dataset yang digunakan untuk studi kasus hanya dimabil pada kurun waktu tertentu. Penelitian selanjutnya diharapkan dapat menggunakan dataset berupa opini yang terbaru secara *realtime*.
b. Pada penelian ini, penggunaan dua model pengujian digunakan untuk melakukan komparasi perbandingan nilai tingkat akurasi. Penelitian selanjutnya diharapkan dapat melakukan kombinasi beberapa model pengujian.
c. Model pengujian yang diterapkan dapat diganti dengan model pengujian lainnya agar lebih variatif.

DAFTAR PUSTAKA

PUSTAKA BUKU

Suyanto. (2018). *Machine Learning Tingkat Dasar dan Lanjut* (Vol. 1). Informatika.

PUSTAKA MAJALAH, JURNAL ILMIAH ATAU PROSIDING

Adhi Putra, A. D. (2021). Analisis Sentimen pada Ulasan pengguna Aplikasi Bibit Dan Bareksa dengan Algoritma KNN. *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, *8*(2), 636–646. https://doi.org/10.35957/jatisi.v8i2.962

Anjas Ramadhan, D., & Budi Setiawan SSi, E. (t.t.). *ANALISIS SENTIMEN PROGRAM ACARA DI SCTV PADA TWITTER MENGGUNAKAN METODE NAIVE BAYES DAN SUPPORT VECTOR MACHINE*.

Arsi, P., Kusuma, B. A., & Nurhakim, A. (2021). Analisis Sentimen Pindah Ibu Kota Berbasis Naive Bayes Classifier. *Jurnal Informatika Upgris*, *7*(1). https://doi.org/10.26877/jiu.v7i1.7636

Asro'i, A., & Februariyanti, H. (2022). ANALISIS SENTIMEN PENGGUNA TWITTER TERHADAP PERPANJANGAN PPKM MENGGUNAKAN METODE K-NEAREST NEIGHBOR. *Jurnal Khatulistiwa Informatika*, *10*(1), 17–24. https://doi.org/10.31294/jki.v10i1.12624

Bayhaqy, A., Sfenrianto, S., Nainggolan, K., & Kaburuan, E. R. (t.t.). *Sentiment Analysis about E-Commerce from Tweets Using Decision Tree, K-Nearest Neighbor, and Naïve Bayes*. http://dlvr.it/Qb83n8pic.twitter.com/8MucIMhUMO,

Darwis, D., Siskawati, N., & Abidin, Z. (t.t.). *Penerapan Algoritma Naive Bayes untuk Analisis Sentimen Review Data Twitter BMKG Nasional*. *15*(1).

Deviyanto, A., Didik Wahyudi, M. R., & Informatika UIN Sunan Kalijaga Yogyakarta Jl Marsda Adi Sucipto No, T. (2018). PENERAPAN ANALISIS SENTIMEN PADA PENGGUNA TWITTER MENGGUNAKAN METODE K-NEAREST NEIGHBOR. *Jurnal Informatika Sunan Kalijaga)*, *3*(1), 1–13. https://twitter.com/search?l=id&q=AHY%20since%3A2017-01-01%20until%3A2017-01-

Duei Putri, D., Nama, G. F., & Sulistiono, W. E. (2022). Analisis Sentimen Kinerja Dewan Perwakilan Rakyat (DPR) Pada Twitter Menggunakan Metode Naive Bayes Classifier. *Jurnal Informatika dan Teknik Elektro Terapan*, *10*(1). https://doi.org/10.23960/jitet.v10i1.2262

Fajar, R., Program, S., Rekayasa, P., Lunak, N., & Bengkalis, R. (t.t.). *Implementasi Algoritma Naive Bayes Terhadap Analisis Sentimen Opini Film Pada Twitter*. *3*(1).

Febriansyah, I., Fikry, M., & Yusra. (2023). Analisis Sentiment di Twitter terhadap Anies Baswedan sebagai Bakal Calon Presiden 2024 Menggunakan Metode K-Nearest Neighbor. *G-Tech: Jurnal Teknologi Terapan*, *7*(3), 1061–1070. https://doi.org/10.33379/gtech.v7i4.2723

Febriyani, E., & Februariyanti, H. (t.t.). *Analisis Sentimen Terhadap Program Kampus Merdeka Menggunakan Algoritma Naive Bayes Classifier Di Twitter*. *17*(1).

Hayuningtyas, R. Y., & Sari, R. (2019). ANALISIS SENTIMEN OPINI PUBLIK BAHASA INDONESIA TERHADAP WISATA TMII MENGGUNAKAN NAÏVE BAYES DAN PSO. *Jurnal TECHNO Nusa Mandiri*, *16*(1), 37. http://nusamandiri.ac.id/

I Komang Andi Sugiarta, Anugrah Cahya Dewi, P., & Nengah Widya Utami. (2023). ANALISA SENTIMEN MAHASISWA TERHADAP LAYANAN STMIK PRIMAKARA MENGGUNAKAN ALGORITMA NAIVE BAYES DAN K-NEAREST NEIGHBOR. *Jurnal Informatika Teknologi dan Sains (Jinteks)*, *5*(3), 364–372. https://doi.org/10.51401/jinteks.v5i3.3159

Isnain, A. R., Sakti, A. I., Alita, D., & Marga, N. S. (2021). SENTIMEN ANALISIS PUBLIK TERHADAP KEBIJAKAN LOCKDOWN PEMERINTAH JAKARTA MENGGUNAKAN ALGORITMA SVM. *Jurnal Data Mining dan Sistem Informasi*, *2*(1), 31. https://doi.org/10.33365/jdmsi.v2i1.1021

Krisandi, N., Helmi, B., & Prihandono, I. (2013). *ALGORITMA k-NEAREST NEIGHBOR DALAM KLASIFIKASI DATA HASIL PRODUKSI KELAPA SAWIT PADA PT. MINAMAS KECAMATAN PARINDU* (Vol. 02, Nomor 1).

Krisdiyanto, T., Maricha, E., & Nurharyanto, O. (2021). Analisis Sentimen Opini Masyarakat Indonesia Terhadap Kebijakan PPKM pada Media Sosial Twitter Menggunakan Naïve Bayes Clasifiers. *Jurnal CoreIT*, *7*(1).

MZ, Y., Edwin Bororing, J., Rahayu, S., & Andhika Putra, J. (2023). Analisis Sentimen Pengguna Apliaksi Shopee Menggunakan Metode Naive Bayes Classifier dan K-NN. *Smart Comp: Jurnalnya Orang Pintar Komputer*, *12*(3), 745–753. https://doi.org/10.30591/smartcomp.v12i3.5494

Naik, A., & Samant, L. (2016). Correlation Review of Classification Algorithm Using Data Mining Tool: WEKA, Rapidminer, Tanagra, Orange and Knime. *Procedia Computer Science*, *85*, 662–668. https://doi.org/10.1016/j.procs.2016.05.251

Ndapamuri, A. M., Manongga, D., & Iriani, A. (2023). Analisis Sentimen Ulasan Aplikasi Tripadvisor Dengan Metode Support Vector Machine, K-Nearest Neighbor, Dan Naive Bayes. *INOVTEK Polbeng - Seri Informatika*, *8*(1), 127. https://doi.org/10.35314/isi.v8i1.3260

Permana, R. A., & Sahara, S. (2023). ALGORITMA K-NEAREST NEIGHBOR PADA ANALISA SENTIMEN REVIEW PRODUK ROUT ER. *Jurnal Sistem Informasi dan Sistem Komputer*, *8*(2). https://doi.org/10.51717/simkom.v8i2.129

Rahayu, S., MZ, Y., Bororing, J. E., & Hadiyat, R. (2022). Implementasi Metode K-Nearest Neighbor (K-NN) untuk

Analisis Sentimen Kepuasan Pengguna Aplikasi Teknologi Finansial FLIP. *Edumatic: Jurnal Pendidikan Informatika*, *6*(1), 98–106. https://doi.org/10.29408/edumatic.v6i1.5433

Salim, S. S., & Mayary, J. (2020). ANALISIS SENTIMEN PENGGUNA TWITTER TERHADAP DOMPET ELEKTRONIK DENGAN METODE LEXICON BASED DAN K – NEAREST NEIGHBOR. *Jurnal Ilmiah Informatika Komputer*, *25*(1), 1–17. https://doi.org/10.35760/ik.2020.v25i1.2411

Saputra, E. F., & Pribadi, M. R. (2023). Analisis Sentimen Komentar Pada Kanal Youtube The Lazy Monday Menggunakan Algoritma Naive Bayes. *MDP Student Conference*, *2*(1), 17–23. https://doi.org/10.35957/mdp-sc.v2i1.4283

Septian, J. A., Fachrudin, T. M., & Nugroho, A. (2019). Analisis Sentimen Pengguna Twitter Terhadap Polemik Persepakbolaan Indonesia Menggunakan Pembobotan TF-IDF dan K-Nearest Neighbor. *Journal of Intelligent System and Computation*, *1*(1), 43–49. https://doi.org/10.52985/insyst.v1i1.36

Supriyanto, J., Korespondensi, P., Alita, D., & Rahman Isnain, A. (t.t.). *Penerapan Algoritma K-Nearest Neighbor (K-NN) Untuk Analisis Sentimen Publik Terhadap Pembelajaran Daring*. *4*, 74–80. https://doi.org/10.33365/jatika.v4i1.2468

Syahrudin, A. N. (2018). *Jurnal Dasar Pemograman Python STMIK*.

Yoga Syantara, A., Dwi Wahyuni, E., & Rahmayanti Setyaning Nastiti, V. (2021). Analisis Sentimen Pada Media Sosial Twitter Menggunakan Naïve Bayes Classifier Terhadap Kata Kunci "#Asiangames2018." *REPOSITOR*, *3*(5), 493–500.

PUSTAKA ELEKTRONIK

Lutfia Afifah. (2020). *Algoritma K-Nearest Neighbor (KNN) untuk Klasifikasi*. https://ilmudatapy.com/algoritma-k-nearest-neighbor-knn-untuk-klasifikasi/

DQLab AI-Powered Learning. (2022). *Mengenal Naive Bayes Sebagai Salah Satu Algoritma Data Science*. https://algorit.ma/blog/naive-bayes-2022/