PERBANDINGAN ALGORITMA KLASIFIKASI *NAÏVE BAYES*, *LOGISTIC REGRESSION*, DAN KNN UNTUK ANALISIS SENTIMEN PEMINATAN MASYARAKAT TERHADAP KANDIDAT BAKAL CALON PRESIDEN 2024

TUGAS AKHIR

Disusun oleh:

ZAKWAN FEBRIAN
NIM. 190705029
Masiswa Fakultas Sains dan Teknologi
Program Studi Teknologi Informasi

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI AR-RANIRY
BANDA ACEH
2023 M / 1444 H

LEMBAR PERSETUJUAN

PERBANDINGAN ALGORITMA KLASIFIKASI *NAÏVE BAYES*, *LOGISTIC REGRESSION*, DAN KNN UNTUK ANALISIS SENTIMEN PEMINATAN MASYARAKAT TERHADAP KANDIDAT BAKAL CALON PRESIDEN 2024

TUGAS AKHIR

Diajukan Kepada Fakultas Sains dan Teknologi
Universitas Islam Negeri (UIN) Ar-Raniry Banda Aceh
Sebagai Salah Satu Beban Studi Memperoleh Gelar Sarjana
pada Prodi Teknologi Informasi

Oleh:
ZAKWAN FEBRIAN
NIM. 190705029
Mahasiswa Fakultas Saius dan Teknologi
Program Studi Teknologi Informasi

Disetujui untuk Dimunaqasyahkan Oleh:

Pembimbing I,	Pembimbing II,
Hendri Ahmadian, M.I.M	Bustami, M.Se.
NIP. 198301042014031002	NIP **198604082014031001**

Mengetahui,
Ketua Program Studi Teknologi Informasi

Ima **Dwitawati, MBA**
NIP. **198210132014032002**

LEMBAR PENGESAHAN

PERBANDINGAN ALGORITMA KLASIFIKASI *NAÏVE BAYES*, *LOGISTIC REGRESSION*, DAN KNN UNTUK ANALISIS SENTIMEN PEMINATAN MASYARAKAT TERHADAP KANDIDAT BAKAL CALON PRESIDEN 2024

TUGAS AKHIR

Telah Diuji Oleh Panitia Ujian Munaqasah Tugas Akhir
Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh dan Dinyatakan Lulus
Serta Diterima Sebagai Salah Satu Studi Program Sarjana (S-1)
Dalam Prodi Teknologi Informasi

Pada Hari/Tanggal: Selasa, 7 November 2023 M
23 Rabiul Akhir 1445 H
di Darussalam, Banda Aceh

Panitia Ujian Munaqasyah Skripsi

Ketua,

Hendri Ahmadian, M. I. M.
NIP. 198301042014031002

Sekretaris,

Bustami, M. Sc.
NIP. 198604082014031010

Penguji I,

Khairan AR, M. Kom.
NIP.198607042014031001

Penguji II,

Mulkan Fadhli, S. T., M. T.
NIP.198607042014031001

Mengetahui
Dekan Fakultas Sains dan Teknologi
UIN Ar-Raniry Banda Aceh,

Dr. Ir. Muhammad Dirhamsyah, M.T., IPU
NIDN. 0002106203

LEMBAR PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama	:	Zakwan Febrian
NIM	:	190705029
Program Studi	:	Teknologi Informasi
Fakultas	:	Sains dan Teknologi
Judul	:	Perbandingan Algoritma *Naïve bayes, Logistic regression,* dan KNN Untuk Analisis Sentimen Peminatan Masyarakat Terhadap Kandidat Bakal Calon Presiden 2024

Dengan ini menyatakan bahwa dalam penulisan tugas akhir ini, saya:

1. Tidak menggunakan ide orang lain tanpa mampu mengembangkan dan mempertanggung jawabkan.
2. Tidak melakukan plagiasi terhadap naskah orang lain.
3. Tidak menggunakan karya orang lain tanpa menyebutkan sumber asli atau tanpa izin pemilik karya.
4. Tidak memanipulasi dan memalsukan data.
5. Mengerjakan sendiri karya ini dan mampu mempertanggungjawab atas karya ini.

Bila kemudian hari ini ada tuntutan dari pihak lain atas karya saya, dan telah melalui pembuktian yang dapat mempertanggung jawabkan dan ternyata memang ditemukan bukti bahwa saya telah melanggar pernyataan ini, maka saya siap dikenakan sanksi berdasarkan aturan yang berlaku di Fakultas Sains dan Teknologi UIN Ar-Raniry Banda Aceh.

Demikian pernyataan ini saya buat dengan sesungguhnya dan tanpa paksaan dari pihak manapun.

Banda Aceh, 21 Desember 2023
Yang Menyatakan

(Zakwan Febrian)

ABSTRAK

Nama	:	Zakwan Febrian
NIM	:	190705029
Program Studi	:	Teknologi Informasi
Fakultas	:	Sains dan Teknologi (FST)
Judul	:	PERBANDINGAN ALGORITMA KLASIFIKASI *NAÏVE BAYES, LOGISTIC REGRESSION,* DAN KNN UNTUK ANALISIS SENTIMEN PEMINATAN MASYARAKAT TERHADAP KANDIDAT BAKAL CALON PRESIDEN 2024
Tanggal Sidang	:	Selasa, 7 November 2023
Pembimbing I	:	Hendri Ahmadian, M. I. M.
Pembimbing II	:	Bustami, M. Sc.

Analisis Sentimen banyak digunakan pemangku kepentingan dalam menilai sentimen terhadap suatu objek. Pada penelitan ini objek yang akan diambil analisisnya yaitu sentimen terhadap tokoh politik bakal calon presiden 2024 yang sedang marak diperbincangkan oleh warganet, khususnya di twitter. Penelitian ini bertujuan untuk menganalisis ukuran kinerja dari penelitian sebelumnya dengan menggunakan algoritma ***Naïve bayes, Logistic regression,* dan KNN**. Penelitian ini mengambil data Twitter yang berhubungan dengan calon presiden untuk melihat opini masyarakat kepada setiap calon presiden. Data yang diambil yaitu data twitter dengan kata kunci **calon presiden** sebanyak **2.201** data yang diambil pada tanggal **18 sampai 25 Januari 2022**. Hasil tersebut menjelaskan terdapat **1471** dataset mengandung positif, **197** dataset mengandung negatif, dan **533** dataset mengandung netral dari pengujian mendapatkan kesimpulan, untuk hasil akurasi algoritma *Logistic regression* mempunyai ukuran kinerja atau akurasi sebesar **0.85%** cukup tinggi jika dibandingkan dengan algoritma *Naïve bayes* dengan nilai akurasi sebesar **0.77%** dan KNN dengan nilai akurasi sebesar **0.71%**.

Kata kunci: Bacapres; Analisis sentimen; *Naïve nayes; Logistic regresion;* KNN.

KATA PENGANTAR

Segala puji dan syukur atas rahmat Allah SWT serta karunia-Nya sehingga dapat menyelesaikan laporan tugas akhir untuk memenuhi syarat nilai mata kuliah. Tak lupa pula, Shalawat beriring salam kepada baginda Rasulullah Shallauhu'alaihiwasalam beserta keluarga dan sahabat beliau yang senantiasa menjunjung tinggi nilai keislaman hingga sampai ini dapat dinikmati oleh setiap manusia.

Hormat saya dan ucapan terima kasih saya kepada setiap pihak yang telah mendukung, baik di dalam materi maupun non materi kepada yang terhormat:

1. Ibuk Ima Dwitawati, MBA. selaku Ketua Prodi Teknologi Informasi.
2. Bapak Hendri Ahmadian, S.Si., M.IM. dan Bapak Bustami, M.Sc. selaku Dosen pembimbing.
3. Semua pihak yang telah membantu dalam penyusunan skripsi yang tidak dapat disebutkan satu persatu.

Penulis menyadari bahwa dalam penulisan karya tulis ini masih jauh dari kesempurnaan serta kesalahan dalam penulisan karya tulis ini masih diluar batas kemampuan penulis. Maka dari itu saran dan kritik yang dibangun dapat diharapkan demi kesempurnaan dalam makalah ini.

Akhir kata, penulis berharap semoga laporan tugas akhir ini berguna bagi para pembaca dan pihak-pihak lain yang berkepentingan.

Banda Aceh, 6 Oktober 2023
Penulis,

Zakwan Febrian

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Euforia dukungan terhadap kandidat bakal Calon Presiden (Bacapres) 2024 mulai terasa sejak tahun 2022 dengan adanya berita yang dilansirkan dari berbagai media yang menjelaskan dukungan masyarakat terhadap tokoh kandidat untuk maju menjadi Calon Presiden 2024. Berdasarkan hasil berita tersebut menjelaskan, terdapat 3 kandidat yang akan maju menjadi Calon Presiden 2024 terdiri dari Anies Baswedan, Ganjar Pranowo, dan Prabowo Subianto.

Dirilis oleh CNN Indonesia pada tanggal 22 Desember 2022 menjelaskan, hasil elektabilitas dukungan tertinggi di peroleh oleh Ganjar Pranowo dengan perolehan sebesar 42.8%, sedangkan untuk Anies Baswedan berada diurutan kedua dengan perolehan 28.1% dan disusul Prabowo Subianto dengan perolehan 23.9%. Hasil elektabilitas dukungan tersebut masih bersifat sementara dikarenakan pada tanggal 3 Mei 2023 dari sumber hasil survei indikator politik menjelaskan, untuk elektabilitas dukungan Prabowo Subianto meningkat menjadi 34.8% sedikit lebih unggul dari Ganjar Pranowo dengan perolehan 34.4% dan Anies Baswedan berada di urutan ketiga dengan perolehan 21.8%.

Hasil sumber informasi elektabilitas dukungan Bacapres dapat dilakukan dengan cara analisis komentar pada media sosial seperti twitter, youtube, instagram, atau media sosial lainnya, teknik tersebut dinamakan sentimen analisis. Sentimen analisis merupakan penentuan nada emosional pada komentar atau cuitan dari penilaian masyarakat terhadap suatu pokok masalah, salah satunya terhadap tokoh Bacapres atau lainnya dengan cara mengelompokkan nada emosional kedalam positif, negatif, atau netral. Dalam pengujian keakuratan klasifikasi bisa dilakukan dengan cara beberapa metode algoritma seperti *Support Vektor Machine* (SVM), *Naïve bayes*, KNN, atau algoritma lainnya.

Beberapa dari penelitian sebelumnya sudah pernah melakukan sebuah analisis sentimen pada kalimat berita di media sosial, salah satunya penelitian tentang penggunaan algoritma SVM dan *Naïve bayes* terhadap berita berkaitan Universitas Muhammadiyah

Malang (Fikri dkk., 2020) [1]. Penelitian ini menjelaskan hasil akurasi metode *Naïve Bayes* dengan nilai 90.62% lebih baik dari metode SVM dengan nilai 90.10%. Adapun penelitian lainnya tentang penggunaan algoritma *Long Short-Term Memory* (LSTM) dan *Naïve Bayes* dalam analisis sentimen terhadap berita Covid-19(Rahman dkk., t.t.) [2]. Penelitian ini menjelaskan hasil akurasi agoritma LSTM dengan nilai 83.33% lebih baik dari *Naïve Bayes* dengan nilai 82%. Hasil penjelasan tersebut bisa disimpulkan perlunya metode-metode lainnya untuk menguji keakuratan pada data tersebut dengan melakukan sebuah perbandingan algoritma dalam analisis sentimen data. Oleh sebab itu, dalam penelitian ini akan melakukan sebuah analisis sentimen terhadap cuitan berita tentang para pendukung Bacapres 2024 dengan menggunakan algoritma *Naïve bayes*, *Logistic regression*, dan KNN serta membandingkan akurasi kinerja pada ketiga algoritma sebagai bahan referensi dalam pembelajaran tersebut.

1.2 Rumusan Masalah

Berdasarkan pada uraian latar belakang, rumusan masalah dari penelitian tugas akhir adalah:

1. Seberapa tingkat kecenderungan pada cuitan twittert terhadap kandidat Bacapres 2024 dari hasil analisis sentimen?
2. Berapa tingkat akurasi dan evaluasi model pada algoritma *naïve bayes*, *logistic regression*, dan KNN dalam proses analisis sentimen komentar twitter terhadap kandidat Bacapres 2024?

1.3 Tujuan Penelitian

1. Menganalisis seberapa banyak komentar positif, negatif, atau netral terhadap kandidat Bacapres 2024 dari hasil analisis sentimen.
2. Mengukur tingkatan akurasi dan evaluasi model pada algoritma *naïve bayes*, *logistic regression*, dan KNN dalam klasifikasi dataset dengan memandingkan metode model lain.

1.4 Batasan Masalah

1. Data yang digunakan adalah hasil dari crawling pada cuitan berita twitter.
2. Model algoritma untuk klasifikasi pada penelitian ini adalah *naïve bayes*, *logistic regression*, dan KNN

BAB II
TINJAUAN PUSTAKA

2.1. Penelitian Terkait

Penelitian dari Ramadhani & Wahyudin (2022) [3] dari Program Studi Sistem Informasi, Fakultas Teknologi Komunikasi dan Informatika, Universitas Nasional tentang "Analisis sentimen terhadap vaksinasi *Astra Zenea* pada twitter menggunakan metode *Naïve bayes* dan KNN" menjelaskan penggunaan algoritma *naïve bayes* dan KNN pada berita vaksinasi *astra zenea* dengan jumlah 407 cuitan dengan perolehan sentimen analisis yang terdiri dari 291 cuitan dengan sentimen netral, 10 cuitan dengan sentimen negatif, dan 105 cuitan dengan sentimen positif menghasilkan perbandingan pengujian dan pengklasifikasian algoritma *naïve bayes* dengan nilai akurasi sebesar 88.56% (*macro avg* 88.62%) lebih tinggi jika dibandingkan dengan algoritma KNN dengan nilai akurasi sebesar 74.78% (*macro avg* 74.77%).

Penelitian dari Assaidi & Amin, t.t.(2020) [4] dari Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Stikubank tentang "Analisis sentimen evaluasi pembelajaran tatap muka 100 persen pada pengguna twitter menggunakan metode *Logistic Regression*" menjelaskan penggunaan algoritma *logistic regression* pada berita pembelajaran tatap muka dengan jumlah 349 cuitan dengan perolehan sentimen analisis yang terdiri dari 177 cuitan dengan sentimen positif dan 172 cuitan dengan sentimen negatif menghasilkan pengujian dan pengklasifikasian algoritma *logistic regression* dengan nilai akurasi sebesar 78.57%, *precision* 76.92%, *recall* 83.92%, dan *f1-score* sebesar 80%.

Penelitian dari Manalu dkk(2022) [5] dari Universitas Methodist Indonesia tentang "Analisis Sentimen twitter terhadap wacana penundaan pemilu dengan metode *Support Vector Machine*" menjelaskan penggunaan *Support Vector Machine* (SVM) pada berita penundaan pemilu 2024 dengan sekitar 100 sampel data. Hasil pengujian SVM pada sampel data tersebut menjelaskan bahwa perolehan nilai negatif dengan sekitar 60 cuitan lebih banyak jika dibandingkan dengan nilai positif dengan sekitar 40 cuitan artinya

masyarakat kurang sependapat tentang wacana penundaan pemilu 2024 sehingga penelitian ini perlu dilakukan pemodelan lainnya untuk proses klasifikasi.

Penelitian dari Fais Sya' bani dkk(2022) [6] tentang "Analisis sentimen terhadap bakal calon presiden 2024 dengan algoritma *Naïve bayes*" menjelaskan penggunaan algoritma *naïve bayes* pada berita yang terkait pada tokoh Bacapres yang terdiri dari:

- tokoh Ganjar Pranowo dengan jumlah 274 cuitan yang terdiri dari 178 cuitan sentimen positif, 54 cuitan sentimen netral, dan 31 cuitan sentimen negatif.
- tokoh Anies Baswedan dengan jumlah 120 cuitan yang terdiri dari 51 cuitan sentimen positif, 16 cuitan sentimen netral, dan 15 cuitan sentimen negatif.
- tokoh Prabowo Subianto dengan jumlah 72 cuitan yang terdiri dari 25 cuitan sentimen positif, 5 cuitan sentimen netral, dan 29 cuitan sentimen negatif.
- dan tokoh Ridwal Kamil dengan jumlah 67 cuitan yang terdiri dari 48 cuitan sentimen positif, 8 cuitan sentimen netral, dan 4 cuitan sentimen negatif.

Menghasilkan pengujian akurasi *naïve bayes* dengan metode *k-fold cross validation* sebesar 73.68% pada iterasi 7 untuk tokoh Ganjar, 71.43% pada iterasi 5 untuk tokoh Anies, 60% untuk tokoh Prabowo, dan 62.5% untuk tokoh Ridwan.

2.2. Landasan Teori

2.2.1. *Machine Learning*

Machine Learning merupakan cabang kecerdasan buatan yang berfokus pada penggunaan data untuk meniru cara kerja manusia. *Machine learning* merupakan komponen terpenting dari bidang ilmu data karena dikhususkan untuk memahami metode belajar dengan memanfaatkan data pelatihan untuk membuat prediksi atau keputusan tanpa diprogramkan secara ekspisit (informasi dengan unsur yang jelas). *Machine learning* terkait dengan berbagai macam seperti statistik komputasi yang berfokus untuk prediksi dengan menggunakan komputer, studi tentang optimasi matematika dengan memberikan teori, dan penambangan data yang berfokus untuk analisis data.

2.2.2. *Artificial Intelligence* (AI)

Artificial intelligence merupakan simulasi proses kecerdasan mesin dengan kemampuan khusus seperti *machine learning*, *deep learning*, *python*, dan *natural language processing* (NLP). Secara umum, AI bekerja dengan menyerap data pelatihan berlabel dengan jumlah yang besar yang diciptakan untuk meniru manusia seperti analisis data, memahami pola, atau membuat suatu keputusan. Perbedaan AI dan komputasi kognitif adalah AI mengacu pada mesin untuk menggantikan kecerdasan manusia dengan cara mensimulasikan terhadap informasi di lingkungan dan komputasi kognitif mengacu pada produk dan layanan dengan meniru proses pemikiran manusia.

2.2.3. Analisis Sentimen

Analisis sentimen merupakan bentuk dalam pengaplikasian text mining dengan tujuan untuk mengetahui opini dari sekumpulan data tentang mengenai peristiwa untuk menentukan apakah nada emosional tersebut mengandung positif, negatif, atau netral. Analisis sentimen merupakan salah satu kunci dari pengembangan *Artificial intelligence* (AI) yang terkait tentang penggalian informasi dari data teks. Dalam dunia bisnis, analisis sentimen sangat diperlukan dalam *role business intelligence* yang bertujuan untuk mengetahui pendapat dari pelanggan tentang mengenai produk atau jasa yang di tawarkan, sehingga bisa digunakan sebagai bahan pertimbangan dalam pengembangan produk. Pada sistem pemerintahan juga dapat membantu untuk mengetahui opini tentang kebijakan yang di keluarkan oleh pemerintah dan memungkinkan untuk melihat perubahan preferensi terhadap publik. Fungsi analisis sentimen untuk media sosial yaitu untuk menggali informasi dan mencari makna pada opini pengguna dari data posting, baik itu *caption* (tulisan pendek pada gambar atau vidio untuk memberikan penjelasan) atau komentar. Proses penggalian analisis sentimen pada media sosial sangat terbatas karena pada saat ekstraksi data untuk menggali sentimen bersifat dinamis, yaitu terdapat pengguna kata yang tidak baku dan format data yang bervariasi.

2.2.4. *K-Fold Cross Validation*

K-fold cross validation merupakan teknik untuk mengevaluasi model prediktif yang dimana pada kumpulan data tersebut dibagi menjadi “k” subset atau lipatan. Model

yang dilatih dan dievaluasi dilakukan sebanyak "k" kali dengan menggunakan lipatan berbeda sebagai set validasi setiap kali. Tujuan metode tersebut yaitu untuk membantu dalam penilaian model, pemilihan, dan penyesuaian *Hyperparameter* sehingga memberikan ukuran efektivitas model yang lebih andal.

Iterasi 1	Test	Training	Training	Training	Training
Iterasi 2	Training	Test	Training	Training	Training
Iterasi 3	Training	Training	Test	Training	Training
Iterasi 4	Training	Training	Training	Test	Training
Iterasi 5	Training	Training	Training	Training	Test

Gambar 2.1. Gambaran *K-fold cross validation*

2.2.5. *Naïve Bayes*

Naïve bayes merupakan algoritma metode klaisfikasi statistik dengan menggunakan teorema *bayes* "*Theorem of Probability*" berdasarkan nilai probalitas (peluang). Metode *naïve bayes* ditemukan oleh ilmuwan penelitian Inggris Thomas Bayes bertujuan untuk memprediksi peluang masa depan dengan berdasarkan pengalaman masa lalu sehingga penelitian ini dinamakan *teorema bayes*. Kombinasi antara teorema dengan *naïve* diasumsi dengan kondisi antribut saling bebas. *naïve bayes* dirancang untuk membuktikan bahwa akurasi dan kecepatan memiliki nilai tinggi saat di input kan kedalam basis data dengan jumlah yang besar.

Proses klasifikasi teks *naïve bayes* dibagi dalam dua tahap yaitu tahap pelatihan dan tahap klasifikasi. Tahap pelatihan dilakukan pada proses sampel data untuk representasi data dengan penetuan probalitas prior (probalitas hipotesis sebelum di amati) berdasarkan sampel data. Tahap klasifikasi ditentukan dengan kategori berdasarkan ketentuan yang muncul dalam bentuk data yang sudah diklasifikasi. Klasifikasi algoritma *naïve bayes* akan dilakukan penentuan nilai probalitas dengan cara melakukan perhitungan jumlah kelas variabel.

$$\mathbf{p(x|i)} = \frac{\boldsymbol{p(i).p(x|i)}}{\sum_j^c = \boldsymbol{p(i).p(x|i)}}$$

Keterangan:

P(x|i) : Probalitas hipotesis x jika fakta

P(i|x) : Mencari nilai parameter kemungkinan yang paling besar

P(x) : *Prior* probaliti dari i

P(i) : Jumlah probaliti yang muncul.

2.2.6. *Logistic Regression*

Regresi logistik biner merupakan algoritma metode analisis digunakan untuk meihat hubungan antara variabel independen dan variabel dependen yang bersifat kategorik (menjelaskan karakteristik pada data) seperti misalnya benar atau salah, hangat atau dingin, dan terindikasi atau tidak terindikasi. Terdapat dua model untuk melakukan analisis yaitu *regresi logistic biner* dan *regresi logistic multinomial*. *Regresi logistic biner* digunakan jika variabel dependen pada data bersifat dikotomi (pemisahan dua bagian). *Regresi logistik multinomial* digunakan jika variabel dependen pada data lebih dari dua kategori.

Metode *regresi logistic biner* merupakan metode yang menggambarkan hubungan variabel independen dengan variabel dependen yang di kategori dengan dua kemungkinan.

$$\mathbf{g(x)} = \boldsymbol{\beta 0 + \beta 1 x 1 + \beta 2 x 2 + \beta 3 x 3 + \beta 4 x 4 A + \beta 5 x 4 B + \beta 6 x 5 + \beta 7 x 6}$$

Keterangan:

g(x) : logit n (x)

β0 : estimasi parameter regresi

β1… β7 : koefisien regresi

x1….x6 : variabel independen.

2.2.7. *K-Nearest Neighbor* (KNN)

K-Nearest Neighbor (KNN) merupakan algoritma klasifikasi objek berfungsi untuk menentukan jarak yang terdekat pada objek antara nilai *K–neighbor* dan data latih berdasarkan hasil dari prediksi. Secara umum, proses algoritma klasifikasi KNN yaitu dengan menentukan jumlah tetangga, menghitug jarak data yang terdekat, dan menentukan hasil dari kelas data. Menghitung jarak antara dua titik bisa menggunakan metode *euclidean distance*.

$$\mathbf{di} = \sqrt{\sum_{i=1}^{n} = (x_{ij} - p_j)^2}$$

Keterangan:

di : Jarak sampel

X_{ij} : Data sampel pengetahuan

P_j : Data input ke – J

n : Jumlah sampel

Algoritma KNN bekerja dengan cara mengambil sejumlah nilai k yang terdekat (tetangga) untuk menentukan nilai kelas dari data baru berdasarkan kemiripan atau kedekatannya pada data yang lain. Hasil dari nilai baru akan diklasifikasi berdasarkan nilai kelas yang sering muncul sebagai nilai prediksi dari hasil pengukuran yang terdekat dengan objek.

Klasifikasi algoritma KNN akan dilakukan penentuan jumlah parameter K (Jumlah tetangga yang dekat) yaitu mempertimbangkan penentuan kelas kategori objek dengan nilai perolehan untuk menyatakan seberapa besarnya data yang berdekatan pada objek. Menghitung jarak tetangga (parameter K) dapat digunakan dengan metode *euclidean distance* dengan cara menghitung kuadrat jarak terdekat data training antara dua variabel. Hasil dari metode *euclidean distance* akan di urutkan secara *ascending* (di urutkan dari nilai tinggi ke rendah) untuk mengklasifikasi berdasarkan nilai k. Hasil kategori terdekat yang paling mayoritas akan dipredisikan sebagai kategori objek.

2.2.8. *Confusion Matrics*

Confusion Matrics merupakan pembelajaran mesin yang di khususkan untuk klasifikasi statistik pada table yang bertujuan untuk mendefinisikan performa algoritma klasifikasi untuk menghasilkan nilai dua output kelas seperti "ya" atau "tidak".

Tabel 2.1. *Confusion matrics*

		Predicted Class		
		Positive	***Neutral***	***Negative***
True Class	***Positive***	*True Positive (TP)*	*False Neutral (FNt)*	*False Negative (FN)*
	Neutral	*False Positive (FP)*	*True Neutral (TNt)*	*False Negative (FN)*
	Negative	*False Positive (FP)*	*False Neutral (FNt)*	*True Negative ((TN)*

Keterangan:

Kondisi positif (P) : Jumlah kasus positif nyata pada data.

Kondisi negative (N) : Jumlah kasus negative nyata pada data.

Positif benar (TP) : Menunjukkan adanya kondisi atau karakteristik.

Negatif benar (TN) : Menunjukkan tidak adanya kondisi atau karakteristik

Netral benar (TN_t) : Tidak menunjukkan postif benar dan negatif benar

Positif palsu (FP) : menunjukkan salah bahwa adanya kondisi atau karakteristik

Negatif palsu (FN) : menunjukkan salah bahwa tidak adanya kondisi atau karakteristik

Netral palsu (FN_t) : Tidak menunjukkan positif palsu dan negatif palsu.

Setiap baris matriks mewakili instance (wujud dari kelas, contoh sebuah rumah di dalamnya ada kursi, tv, dan lain-lain. Rumah adalah kelas dan kursi, tv, dan lain-lain adalah instance) sedangkan setiap kolom mewakili instance yang akan di prediksi.

2.2.9. *Accuracy*

Accuracy merupakan hasil pengujian berdasarkan tingkat perdekatan antara nilai aktual dengan nilai prediksi.

$$\textbf{Accuracy} = \frac{\boldsymbol{TP + TNt + TN}}{\boldsymbol{TP + FP + TNt + FNt + TN + FN}}$$

2.2.10 *Precision*

Precision merupakan perhitungan hasil perbandingan antara jumlah informasi yang didapat dengan seluruh informasi yang diambil.

$$\textbf{Precision}_{\boldsymbol{Positive}} = \frac{\boldsymbol{TP}}{\boldsymbol{TP + FP}}$$

2.2.11 *Recall*

Recall merupakan perhitungan hasil pengujian dengan membandingkan jumlah informasi yang didapat dengan seluruh informasi yang ada pada koleksi informasi (baik yang sudah terambil atau tidak terambil oleh sistem). Perbedaan perumusan *Precission* dan *Recall* adalah perhitungan *False positive* dilakukan pada *precission* dan perhitungan *False negative* dilakukan pada *Recall*.

$$\boldsymbol{Recall_{Positive}} = \frac{\boldsymbol{TP}}{\boldsymbol{TP + FNt + FN}}$$

2.2.12 *F1-score*

F1-score merupakan pengukuran akurasi tes yang dihitung dari hasil presisi dengan perolehan tes untuk mengklasifikasi angka biner. *F1-score* juga berfungsi sebagai metrik untuk nilai rata - rata antara *Precision* dan *Recall* yang mana pada hasil tersebut akan digunakan dalam laporan klasifikasi.

$$\boldsymbol{F1 - score = \frac{Precision * Recall}{Precision + Recall}}$$

2.2.13 *Macro Average*

Macro average merupakan perhitungan skor rata – rata aritmatika dari semua skor per kelas. Metode ini memperlakukan semua kelas secara setara terlepas dari nilai dukungan.

$$\boldsymbol{Macro\ Avg = \frac{\text{Positif} + \text{Negatif} + \text{Netral}}{\text{N}}}$$

Keterangan:

N : Jumlah per kelas

Tabel 2.2. *Macro Avg F1-score*

Label	***F1 - score***	***Macro Avg F1-score***
Positif	*F1 score* P	$\frac{\boldsymbol{F1\ score\ \text{P} + F1\ score\ \text{N} + F1\ score\ \text{Nt}}}{\boldsymbol{3}}$ = *Macro Avg F1-score*
Negatif	*F1 score* N	
Netral	*F1 score* Nt	

2.2.14 *Weighted Average*

Weighted average merupakan hitungan rata – rata tertimbang dengan mengambil rata – rata seluruh skor per kelas dengan mempertimbangkan dukungan masing – masing kelas. Dukungan masing – masing kelas tersebut mengacu pada jumlah kemunculan aktual kelas dalam kumpulan data. misalnya, nilai *Support* 1 pada label positif berarti hanya ada satu observasi dengan label positif yang sebenarnya.

$$\boldsymbol{Weighted\ Avg} = (\mathbf{P} * \boldsymbol{Support}\ \mathbf{P}) + (\mathbf{N} * \boldsymbol{Support}\ \mathbf{N}) + (\mathbf{Nt} * \boldsymbol{Support}\ \mathbf{Nt})$$

Keterangan:

Support : jumlah isi dalam kelas

P : Positif

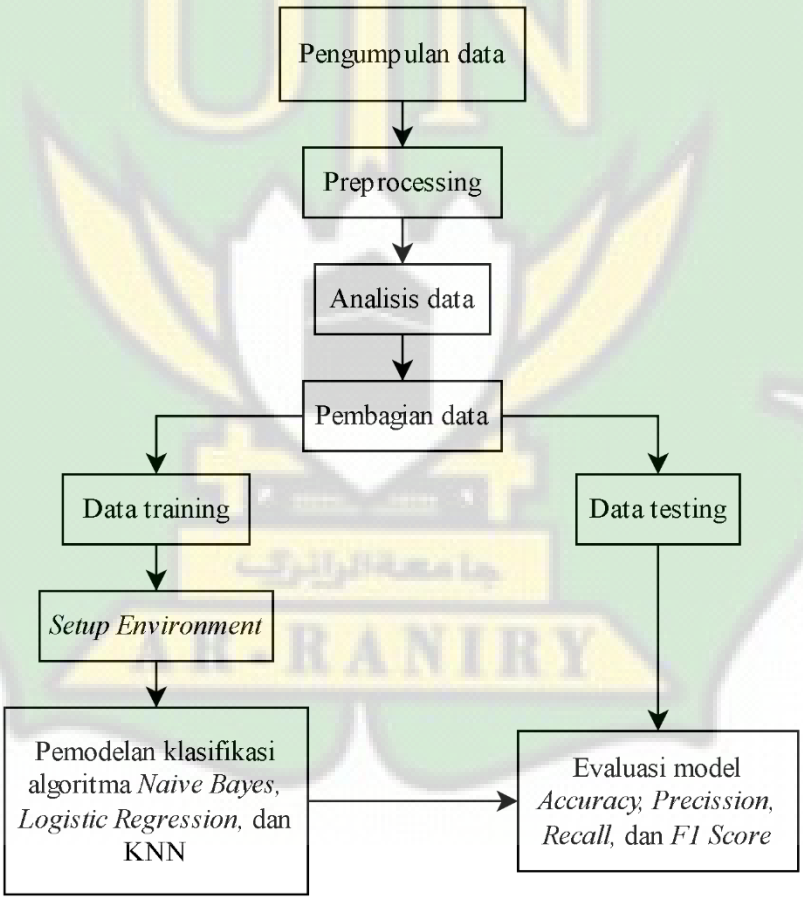
N : Negatif

Nt : Netral

BAB III
METODE PENELITIAN

3.1. Tahap penelitian

Penelitian ini menggunakan penelitian kuantitatif, yaitu melakukan pengumpulan data dan data tersebut akan dilakukan pembersihan data sebelum dilakukan ke tahap sentimen analisis. Setelah dilakukan ke tahap sentimen analisis, data tersebut akan diklasifikasi kedalam algoritma dan akan dibandingkan lagi kedalam akurasi dan evaluasi model. Berikut tahapan penelitian dapat dilihat pada gambar 3.1.



Gambar 3.1. Tahap Penelitian

3.2. *Tools* yang digunakan

Dalam penelitian ini, *Tools* yang digunakan dalam sentimen analisis dan pengklasifikasi algoritma *naïve bayes, logistic regression,* dan KNN menggunakan bahasa pemograman *Python* dengan *compiler* Google colab.

3.3. Pengumpulan data

Proses pengumpulan data menggunakan metode teknik crawling data twitter implementasi dari *library twint* dengan kata kunci "calon presiden". Proses pengumpulan data tersebut dimulai dari tanggal 18 Januari 2023 sampai 25 Januari 2023 dengan hasil data yang sudah terkumpul berjumlah 2.201 dataset. Selanjutnya, dataset yang sudah terkumpul akan disimpan kedalam *File comma separated* (CSV) untuk memungkinkan data tersebut disimpan ke dalam format tabel terstruktur.

3.4. *Preprocessing*

Preprocesing tersebut bertujuan untuk pembersihan data dari data atribut yang tidak terstruktur. Tahap pembersihan data tersebut terdiri dari *Case Folding, Cleansing, Tokenizing,* dan *Stopword Removal***.**

a. *Case Folding*

Case folding merupakan sebuah proses yang menerapkan pada serangkaian karakter, dimana karakter yang diidentifikasi sebagai non-huruf kecil diganti dengan huruf kecil yang setara. Adapun contoh dari *case folding* bisa dilihat pada tabel 3.1.

Tabel 3.1. *Case folding*

Case Folding	
Teks	**Teks Output**
Saya suka video MATA NAJWA Memilih Wakil Rakyat Iwan Fals Ahok JK DS Full	saya suka video mata najwa memilih wakil rakyat iwan fals ahok jk ds full

```
df_penerima['tweet'] = df_penerima['tweet'].str.lower()

print('Case Folding Result : \n')
print(df_penerima['tweet'].head(5))
print('\n\n\n')
```

```
Case Folding Result :

0    @tan_mar3m tak satukatapun terucap , yang dulu...
1          @gomugomuno0 @kritongg penting 7 ucl xixixi
2    selangkah lagi puan maharani ditetapkan sebaga...
3    puan maharani berpeluang besar menjadi calon p...
4    @jokowi pak jangan ikut-ikutan menyalakan api ...
Name: tweet, dtype: object
```

Gambar 3.2. *Case folding*

b. *Cleansing*

Cleansing merupakan proses pembersihan data yang tidak relevan, duplikat, atau tidak terformat seperti URL, hastag (#), username (@username), atau kata frasa lainnya. Berikut contoh dari *cleansing* bisa dilihat pada tabel 3.2.

Tabel 3.2. *Cleansing*

Cleansing	
Teks	**Teks Output**
stand up comedy anang! stand up comedy anang lucu banget terbaru 2014 indonesia metro tv http://www.youtube.com/watch?v=JoSg0a4Q8vY	stand up comedy anang stand up comedy anang lucu banget terbaru 2014 indonesia metro tv

```
def remove_tweet_special(text):
  text = text.replace('\\t'," ").replace('\\n'," ").replace('\\u'," ").replace('\\',"")
  text = text.encode('ascii', 'replace').decode('ascii')
  text = ' '.join(re.sub("([@#][A-Aa-z0-9]+)|(\w+:\/\/\S+)"," ", text).split())
  return text.replace("http://", " ").replace("https://", " ")
df_penerima['tweet'] = df_penerima['tweet'].apply(remove_tweet_special)

def remove_number(text):
  return re.sub(r"\d", "", text)
df_penerima['tweet'] = df_penerima['tweet'].apply(remove_number)

def remove_punctuation(text):
  return text.translate(str.maketrans("","",string.punctuation))
df_penerima['tweet'] = df_penerima['tweet'].apply(remove_punctuation)

def remove_whitespace_LT(text):
  return text.strip()
df_penerima['tweet'] = df_penerima['tweet'].apply(remove_whitespace_LT)

def remove_whitespace_multiple(text):
  return re.sub('\s',' ',text)
df_penerima['tweet'] = df_penerima['tweet'].apply(remove_whitespace_multiple)

def remove_singl_char(text):
  return re.sub(r"\b[a-zA-Z]\b","",text)
df_penerima['tweet'] = df_penerima['tweet'].apply(remove_singl_char)

def word_tokenize_wrapper(text):
  return word_tokenize(text)
df_penerima['tweet'] = df_penerima['tweet'].apply(word_tokenize_wrapper)
```

Gambar 3.3. *Cleansing*

c. *Tokenizing*

Tokenizing merupakan metode pemisahan kata dalam suatu kalimat dengan tujuan untuk proses analisis sentimen. Berikut contoh dari *tokenizing* bisa dilihat pada tabel 3.3.

Tabel 3.3. *Tokenizing*

Tokenizing	
Teks	**Teks Output**
stand up comedy anang lucu banget terbaru 2014 indonesia metro tv	[‘stand’, ‘up’, ‘comedy’, ‘anang’, ‘lucu’, ‘banget’, ‘terbaru’, ‘2014’, ‘indonesia’, ‘metro’, ‘tv’]

```
[ ]  def remove_singl_char(text):
       return re.sub(r"\b[a-zA-Z]\b","",text)
     df_penerima['tweet'] = df_penerima['tweet'].apply(remove_singl_char)

     def word_tokenize_wrapper(text):
       return word_tokenize(text)
     df_penerima['tweet'] = df_penerima['tweet'].apply(word_tokenize_wrapper)

[ ]  print('Tokenizing Result : \n')
     print(df_penerima['tweet'].head(10))
     print('\n\n\n')

     Tokenizing Result :

     0    [marm, tak, satukatapun, terucap, yang, dulu, ...
     1                                [penting, ucl, xixixi]
     2    [selangkah, lagi, puan, maharani, ditetapkan, ...
     3    [puan, maharani, berpeluang, besar, menjadi, c...
     4    [pak, jangan, ikutikutan, menyalakan, api, ya,...
     5    [sementara, itu, ketua, dpp, ppp, achmad, baid...
     6    [artai, golkar, menegaskan, tidak, ada, peruba...
     7    [dilirik, ppp, berkat, kinerja, menteri, bumn,...
     8    [dengan, gabung, ke, golkar, posisi, ridwan, k...
     9    [peluang, menteri, badan, usaha, milik, negara...
     Name: tweet, dtype: object
```

Gambar 3.4. *Tokenizing*

d. *Stopword removal*

Merupakan proses pembuangan kata-kata yang biasanya sering muncul dan dianggap tidak memiliki makna contohnya adalah “sebuah”, “di”,” yang”,” itu”, dan lain-lain. Berikut contoh dari *stopword removal* bisa dilihat pada tabel 3.4.

Tabel 3.4. *Stopword removal*

Stopword Removal	
Teks	**Teks Output**
[‘stand’, ‘up’, ‘comedy’, ‘anang’, ‘lucu’, ‘banget’, ‘terbaru’, ‘di’, ‘indonesia’, ‘metro’, ‘tv’]	[‘stand’, ‘up’, ‘comedy’, ‘anang’, ‘lucu’, ‘banget’, ‘terbaru’, ‘indonesia’, ‘metro’, ‘tv’]

```
[ ] list_stopwords = stopwords.words('indonesian')
    list_stopwords.extend(['yg','dg','rt','dgn','ny','d','klo',
                           'kalo','amp','blar','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','xixixi','ucl'])

    txt_stopword = pd.read_csv("calon_presiden.csv", names = ["stopwords"], header =None)
    list_stopwords.extend(txt_stopword["stopwords"][0].split(' '))
    list_stopwords = set(list_stopwords)

    def stopwords_removal(words):
      return [word for word in words if word not in list_stopwords]
    df_penerima['tweet_tokens_wsw'] = df_penerima['tweet'].apply(stopwords_removal)

[ ] print(df_penerima['tweet_tokens_wsw'].head(10))
```

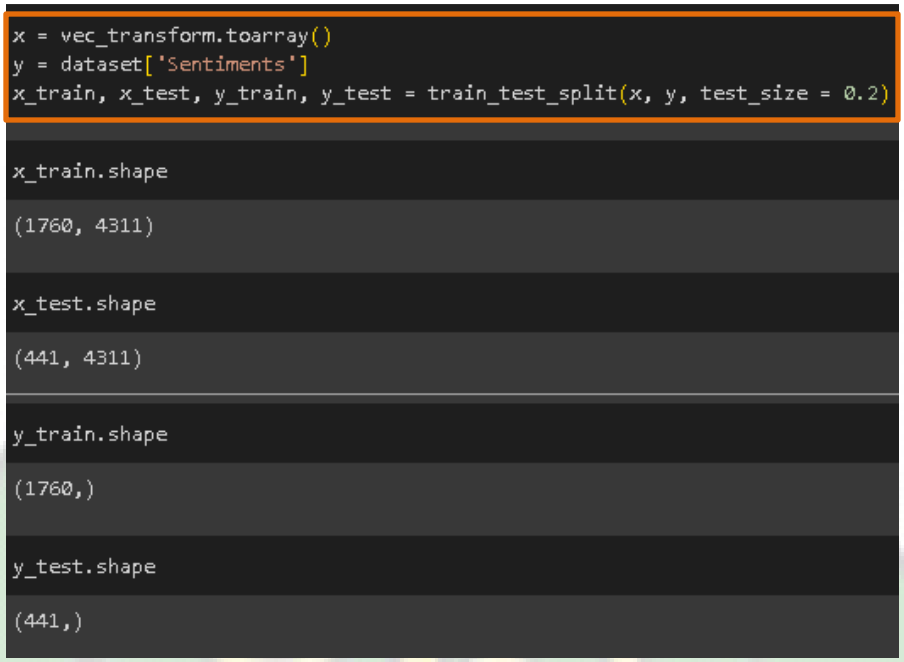
Gambar 3.5. *Stopword removal*

3.5. Analisis sentimen

Tahap analisis sentimen merupakan proses pengelompokan data positif, negatif, dan netral dengan melakukan pengekstrakan kata teks menjadi numerik untuk dilakukan perhintungan ke dalam sentimen analisis. Setiap nilai dari hasil sentimen analisis pada data tersebut akan menggunakan kode angka sebagai pengelompokan data dengan ketentuan 2 sebagai positif, 1 sebagai netral, dan 0 sebagai negatif. Pelabelan positif (kode = 2) menunjukkan bahwa hasil pada cuitan komentar tersebut terdapat pujian atau dukungan terhadap kandidat capres 2024. sedangkan untuk kalimat cacian, hinaan dan ejekan disebut golongan negatif. Dan jika tidak keduanya, maka data tersebut digolongkan kedalam netral.

3.6. Pembagian data

Pembagian data dibagi kedalam 2 terdiri dari data *training* sebagai melatih model dan data *testing* sebagai penguji model. Pembagian data tersebut dilakukan dengan perbandingan 80:20 yaitu, 80% untuk data *training* dan 20% untuk data *testing*. Pembagian data tersebut menggunakan metode *hold – out* yaitu dengan sebagian besar 20% diambil sebagai data *testing* dan sisanya digunakan sebagai data *training*.



Gambar 3.6. Pembagian data *training* dan data *testing*

3.7. Klasifikasi pemodelan algoritma

Pada tahap ini, data tersebut akan dilakukan pemodelan algoritma *Naïve Bayes*, *Logistic Regression*, dan KNN. Adapun penjelasan pengklasifikasian algoritma tersebut sebagai berikut:

a. Klasifikasi pemodelan algoritma *naïve bayes*

Pengklasifikasi algoritma *naïve bayes* menggunakan model *multinomialNB* implementasi dari *library sklearn naive_bayes* karena digunakan untuk klasifkasi data diskrit (data yang nilainya adalah bilangan asli, bukan berupa pecahan angka. Misalnya data berat badan mahasiswa, data jumlah kendaraan, atau lainnya.). *MultinomialNB* merupakan salah satu dari dua varian *Naïve bayes* klasik yang digunakan dalam klasifikasi teks (biasanya data tersebut direpresentasikan sebagai jumlah vektor kata) yang distribusi oleh vektor $\theta_y = (\theta_{y1} \ldots, \theta_{yn})$ untuk setiap kelas y, dimana n merupakan jumlah klasifikasi

teks atau ukuran kosakata, dan θ_{yi} merupakan fitur yang muncul dalam sampel milik kelas *y*. Probalitas suatu kelas *y* dapat dihitung menggunakan persamaan.

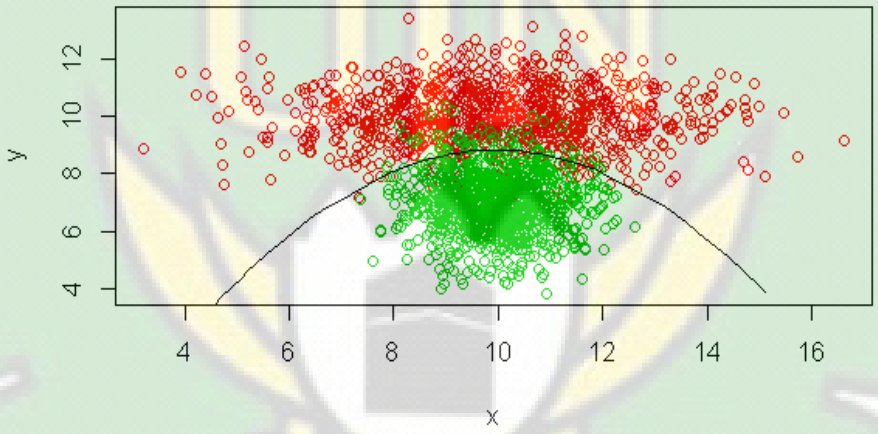
$$\boldsymbol{\theta_{yi} = \frac{N_{yi}}{N_y}}$$

Keterangan:

θ_{yi} : Probalitas

N_{yi} : Fitur berapa kali *i* muncul dalam kelas *y* dalam set pelatihan.

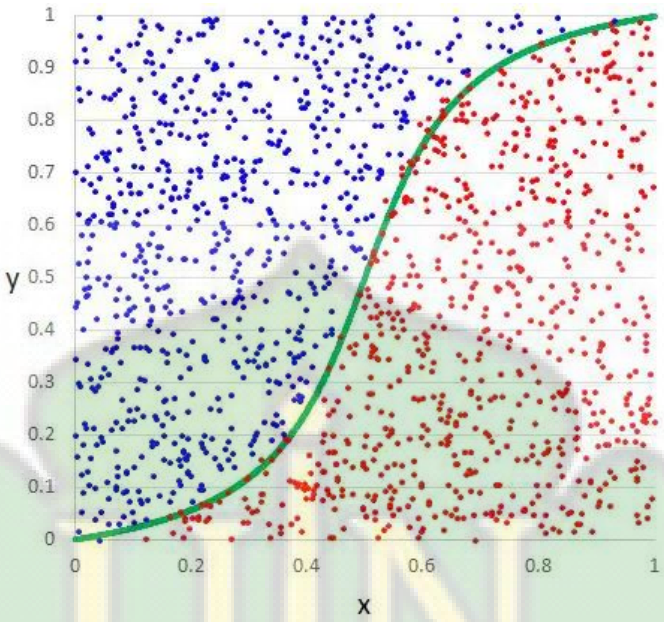
N_y : Jumlah total semua fitur untuk kelas *y*.



Gambar 3.7. Gambaran klasifikasi *naïve bayes*

b. Klasifikasi pemodelan algoritma *logistic regression*

Pengklasifikasi algoritma logistic regression menggunakan model *linear* implementasi dari *library sklearn linear_model* karena digunakan untuk menentukan perkiraan hubungan linear antar variabel dependen (variabel terikat atau sumbu y) dan variabel independen (variabel bebas atau sumbu x) yang dibatasin dengan sebuah garis *regresi*.



Gambar 3.8. Gambaran klasifikasi *logistic regression*

Penerapan hubungan linear model antara variabel dependen Y dan variabel independen X ditentukan dengan pesamaan jumlah prediktor.

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_r x_r + \epsilon$$

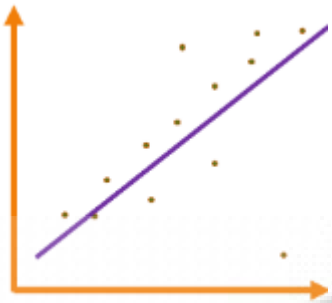
Keterangan:

r : jumlah prediktor

$\beta_{1,\ldots,}\ \beta_r$: koefisien regresi

ϵ : kesalahan acak

Dalam penilaian positif dan negatif pada model regresi sederhana, koefisien bernilai positif jika nilai sumbu X bertambah dan nilai sumbu Y juga bertambah dan koefisien bernilai negatif jika nilai sumbu X bertambah dan nilai Y berkurang.

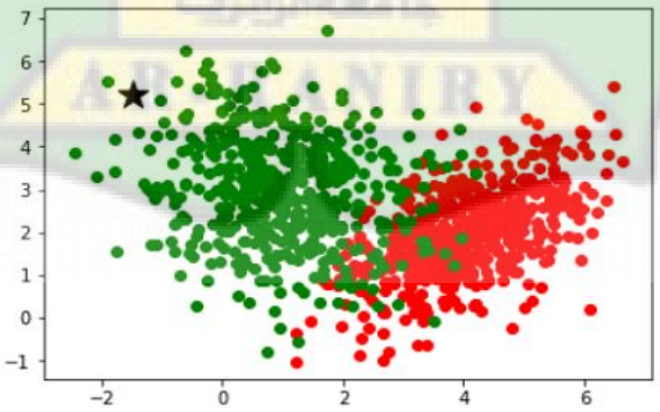


Gambar 3.9. koefisien positif

Gambar 3.10. koefisien negatif

c. Klasifikasi pemodelan algoritma KNN

Pengklasifikasi algoritma KNN menggunakan model *neighbor* implementasi dari *library sklearn neighbors* karena digunakan untuk menemukan titik data yang paling dekat dengan titik yang perlu diprediksi dengan menggunakan jarak *euclidean* (perhitungan jarak antara dua benda yang bukan titik biasanya didefinisikan sebagai jarak terkecil antara pasangan titik dari kedua benda) dan KKN adalah contoh model nonlinier karena menggunakan pendekatan apapun tanpa menggunakan garis untuk memisahkan kasus. Klasifikasi algoritma KNN kedalam positif dan negatif dilakukan perhitungan jarak *euclidean* untuk menghitung jarak antara data yang di evaluasi dengan semua data pelatihan (*training*) dan hasil dari perhitungan jarak *euclidean* di lakukan pengurutan jarak terdekat kedalam nilai positif dan negatif.



Gambar 3.11. Gambaran klasifikasi KNN

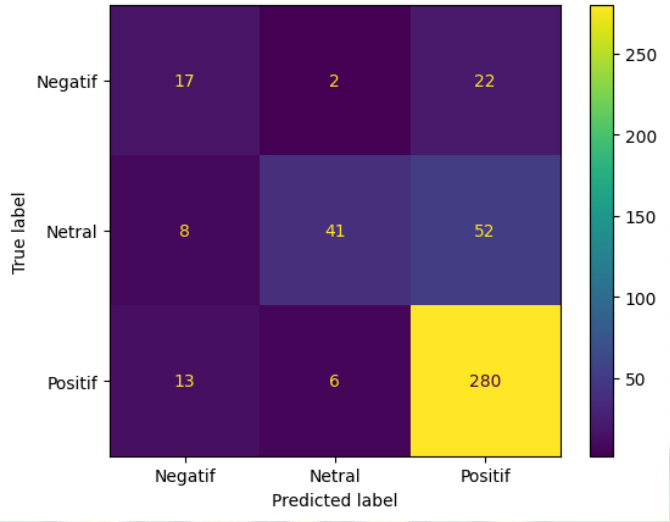
3.8. Hyperparameter model

Tahap Hyperparameter model tersebut merupakan tahap yang digunakan untuk mengukur parameter akurasi pada algoritma *Naïve Bayes, Logistic Regression*, dan KNN untuk menilai performa model pembelajaran mesin dan memperkirakan kemampuan generalisasinya dengan menggunakan metode *K-fold cross validation*. Berikut langkah – langkah *K-fold cross validation* sebagai berikut:

a. **Tahap pemisah data:** tahap pemisah data dilakukan apabila jika nilai "k" di pilih sebagai 5 atau sesuai kebutuhan, maka data tersebut akan di bagi sesuai nilai "k". Contohnya, jika pada sebuah data tersebut akan menggunakan nilai k-5, maka data tersebut akan di bagi menjadi 5 bagian data.
b. **Tahap latih dan validasi:** setiap lipatan iterasi k-1 digunakan untuk melatih model dan lipatan sisanya digunakan sebagai validasi (evaluasi performa).
c. **Tahap kinerja metrik:** hasil dari latih dan validasi pada iterasi tersebut akan dilakukan pengukuran akurasi pada tiap iterasi tersebut.
d. **Tahap nilai rata – rata:** nilai dari setiap iterasi akan dilakukan nilai rata – rata untuk mewakili performa model secara keseluruhan.
e. **Tahap pemilihan dan penyesuaian model:** berdasarkan hasil validasi silang tersebut akan dibandingkan model atau pengaturan *Hyperparameter* untuk memilih salah satu yang memiliki nilai rata – rata yang terbaik pada model tersebut.
f. **Tahap evaluasi akhir:** setelah memilih model atau *Hyperparameter*, akan dilakukan pengujian ulang kedalam evaluasi model secara terpisah untuk mendapatkan estimasi performa akhir.

3.9. Evaluasi model

Setelah dilakukan penerapan algoritma *naïve bayes*, *logistic regression*, dan KNN, akan dilakukan ke tahap evaluasi model yang terdiri dari 2 tahapan yaitu, tahap *confusion matrics* untuk hasil klasifikasi algoritma dan pengukuran performa pada algoritma dengan melakukan perhitungan akurasi, *Precission, Recall, F1 score, Macro avg,* dan *Weighted avg*.



Gambar 3.12. *Confusion matrics*

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
Negatif	0.76	0.39	0.52	41
Netral	0.46	0.97	0.62	101
Positif	0.97	0.67	0.79	299
accuracy			0.71	441
macro avg	0.73	0.68	0.64	441
weighted avg	0.83	0.71	0.73	441

Gambar 3.13. Pengukuran performa

Hasil dari pengujian tersebut akan dibandingkan untuk melihat nilai tingkat akurasi dan kinerja pada ketiga model algoritma tersebut.

BAB IV
HASIL DAN PEMBAHASAN

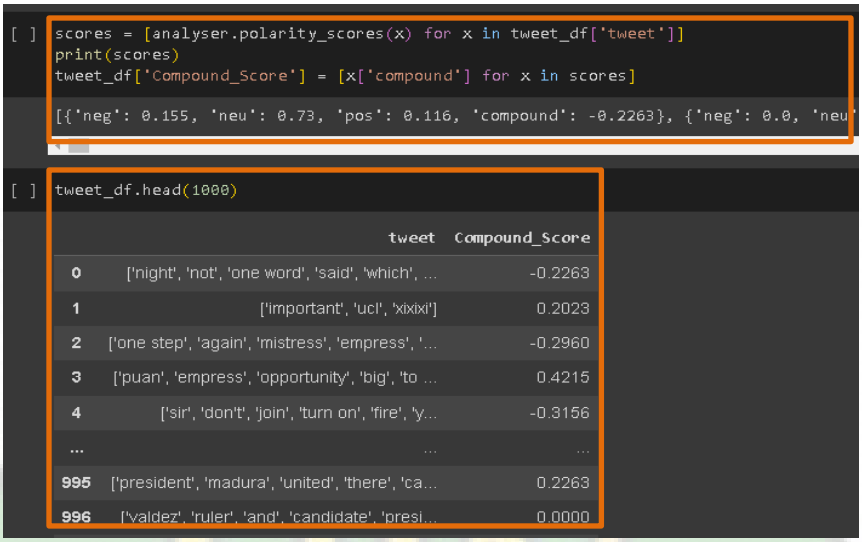
4.1. Deskripsi Analisis Sentimen Data

Penelitian ini, data dengan jumlah 2201 dataset akan dilakukan ke tahap sentimen analisis dengan menggunakan metode *Vader Sentimen* implementasi dari *library VaderSentiment*.

```
#Lexicion Based
!pip install VaderSentiment

Collecting VaderSentiment
  Downloading vaderSentiment-3.3.2-py2.py3-none-any.whl (125 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 126.0/126.0 kB 2.6 MB/s eta 0:00:00
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (fr
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/d
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-pa
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-pa
Installing collected packages: VaderSentiment
Successfully installed VaderSentiment-3.3.2

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
analyser = SentimentIntensityAnalyzer()

scores = [analyser.polarity_scores(x) for x in tweet_df['tweet']]
print(scores)
tweet_df['Compound_Score'] = [x['compound'] for x in scores]

[{'neg': 0.155, 'neu': 0.73, 'pos': 0.116, 'compound': -0.2263}, {'neg': 0.0, 'neu': 0
```

Gambar 4.1. Implementasi data ke dalam *library VaderSentiment*.

Vader Sentiment tersebut bertujuan sebagai sentimen analisis digunakan sebagai pengukur nada emosional pada data teks untuk mendapatkan nilai keluaran hasil disebut sebagai nilai *Compound score*.

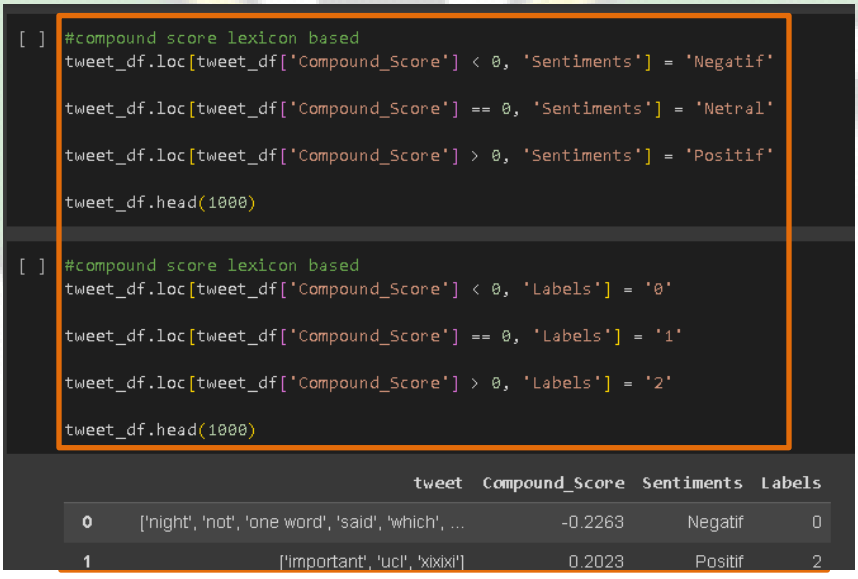


```
[ ] scores = [analyser.polarity_scores(x) for x in tweet_df['tweet']]
    print(scores)
    tweet_df['Compound_Score'] = [x['compound'] for x in scores]

    [{'neg': 0.155, 'neu': 0.73, 'pos': 0.116, 'compound': -0.2263}, {'neg': 0.0, 'neu'

[ ] tweet_df.head(1000)
```

	tweet	Compound_Score
0	['night', 'not', 'one word', 'said', 'which', ...	-0.2263
1	['important', 'ucl', 'xixixi']	0.2023
2	['one step', 'again', 'mistress', 'empress', '...	-0.2960
3	['puan', 'empress', 'opportunity', 'big', 'to ...	0.4215
4	['sir', 'don't', 'join', 'turn on', 'fire', 'y...	-0.3156
...	...	...
995	['president', 'madura', 'united', 'there', 'ca...	0.2263
996	['valdez', 'ruler', 'and', 'candidate', 'presi...	0.0000

Gambar 4.2. *Compound score*

Selanjutnya, hasil dari nilai *Compound score* tersebut akan dilakukan ketahap pengelompokan data dengan cara, apabila jika hasil keluaran tersebut bernilai diatas dari nilai 0, maka data tersebut dikelompokkan kedalam positif. Sedangkan, jika hasil keluaran tersebut bernilai dibawah dari nilai 0, maka data tersebut dikelompokkan kedalam negatif. dan jika untuk tidak kedua nya, maka data tersebut dikelompokan kedalam netral.



```
[ ] #compound score lexicon based
    tweet_df.loc[tweet_df['Compound_Score'] < 0, 'Sentiments'] = 'Negatif'

    tweet_df.loc[tweet_df['Compound_Score'] == 0, 'Sentiments'] = 'Netral'

    tweet_df.loc[tweet_df['Compound_Score'] > 0, 'Sentiments'] = 'Positif'

    tweet_df.head(1000)

[ ] #compound score lexicon based
    tweet_df.loc[tweet_df['Compound_Score'] < 0, 'Labels'] = '0'

    tweet_df.loc[tweet_df['Compound_Score'] == 0, 'Labels'] = '1'

    tweet_df.loc[tweet_df['Compound_Score'] > 0, 'Labels'] = '2'

    tweet_df.head(1000)
```

	tweet	Compound_Score	Sentiments	Labels
0	['night', 'not', 'one word', 'said', 'which', ...	-0.2263	Negatif	0
1	['important', 'ucl', 'xixixi']	0.2023	Positif	2

Gambar 4.3. Pelabelan data

Selanjutnya perolehan data dari hasil analisis sentimen tersebut dengan melakukan perhitungan *value_counts* menjelaskan, jumlah data yang dikelompokan kedalam positif menghasilkan perolehan sebesar 1471 dataset.

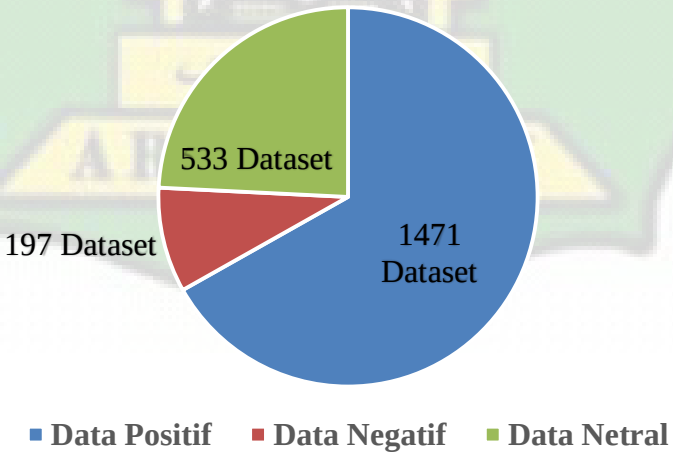
```
dataset['Sentiments'].value_counts()

Positif    1471
Netral      533
Negatif     197
Name: Sentiments, dtype: int64
```

Gambar 4.4. Perhitungan *value_counts*

Selanjutnya untuk data yang dikelompokan kedalam negatif menghasilkan perolehan sebesar 197 dataset. Dan yang terakhir untuk data yang dikelompokan kedalam netral menghasilkan perolehan sebesar 533 dataset.

Tabel 4.1. Jumlah Analisis Sentimen dataset

Kata Kunci Twitter	Jumlah Dataset	Analisis Sentimen		
		Positif	Negatif	Netral
Calon Presiden	2201 dataset	1471 dataset	197 dataset	533 dataset



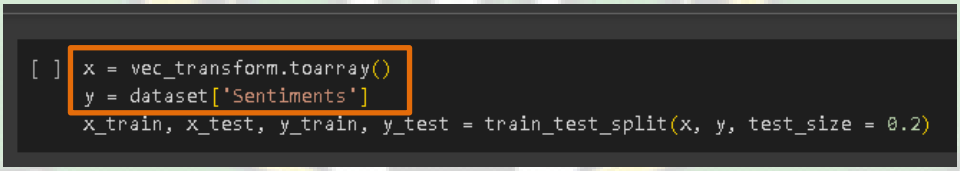
Gambar 4.5. Diagram lingkaran analisis sentimen dataset

4.2. Hyperparameter Model

Penelitian selanjutnya yaitu menentukan nilai probalitas keakuratan pada data analisis sentimen dengan menggunakan Algoritma *Naïve bayes, Logistic* regression, dan KNN. Dalam pemodelan tersebut akan meliputi kedalam tiap – tiap fase yang terdiri dari fase penerapan modeling data dan fase pemodelan parameter algoritma dengan menggunakan teknik *K-Fold Cross Validation*.

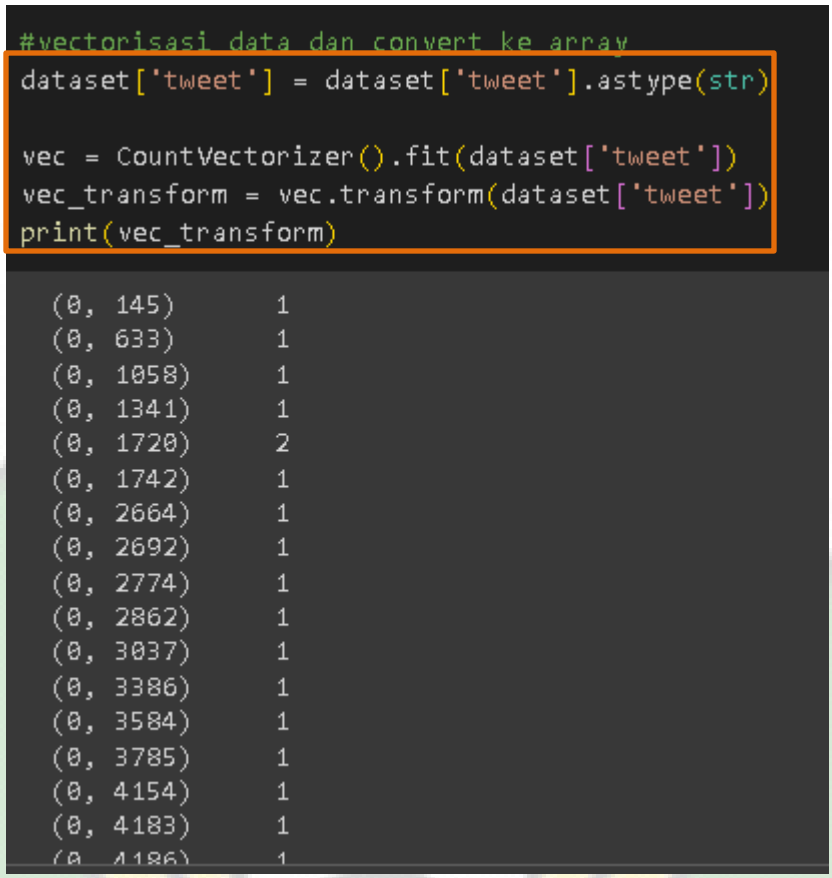
a. Penerapan Modeling Data

Pada fase ini, data yang berjumlah 2201 dataset akan dilakukan penerapan modeling data ke dalam 2 kategori yang terdiri dari **data x** yang merupakan data teks yang bertujuan agar data tersebut dapat terbaca ketika menjalankan kedalam model algoritma *naïve bayes, logistic regression*, dan KNN dalam pelatihan atau pengujian keakuratan pada data sentimen analisis dan **data y** yang merupakan data atribut sentimen analisis.



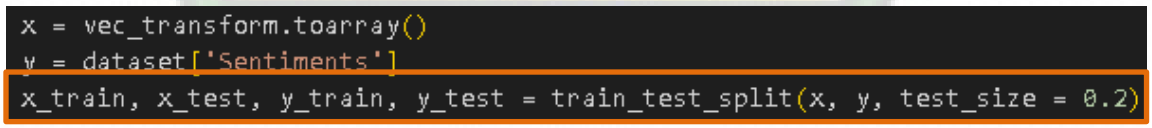
Gambar 4.6. Penerapan modeling data

Proses penerapan modeling data tersebut dilakukan pada saat pengonversian data tipe objek ke dalam *Count vectorizer* implementasi dari *library sklearn feature_extraction text* sehingga hasil dari implementasi tersebut menghasilkan data bilangan metriks. *Count vectorizer* merupakan alat yang digunakan untuk mengubah fitur teks menjadi bilangan metriks untuk ditugaskan sebagai penerjemah dalam pembelajaran bahasa mesin dan pemrosesan bahasa alami (NLP). hasil implementasi tersebut akan di inputkan ke dalam **data x.**



Gambar 4.7. Mengonversi data tipe objek ke dalam matriks

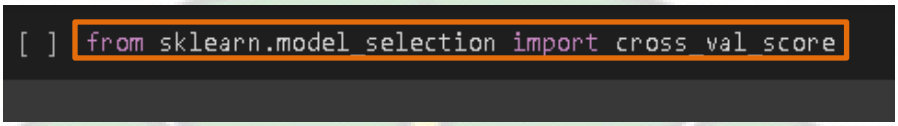
Selanjutnya, penerapan modeling data tersebut akan dilakukan pembagian data ke dalam data *training* sebagai pelatihan model algoritma dan data *testing* sebagai pengujian akurasi model dengan perbandingan 80:20 yaitu, 1760 dataset untuk data *training* dan 441 dataset untuk data *testing*.



Gambar 4.8. Pembagian data

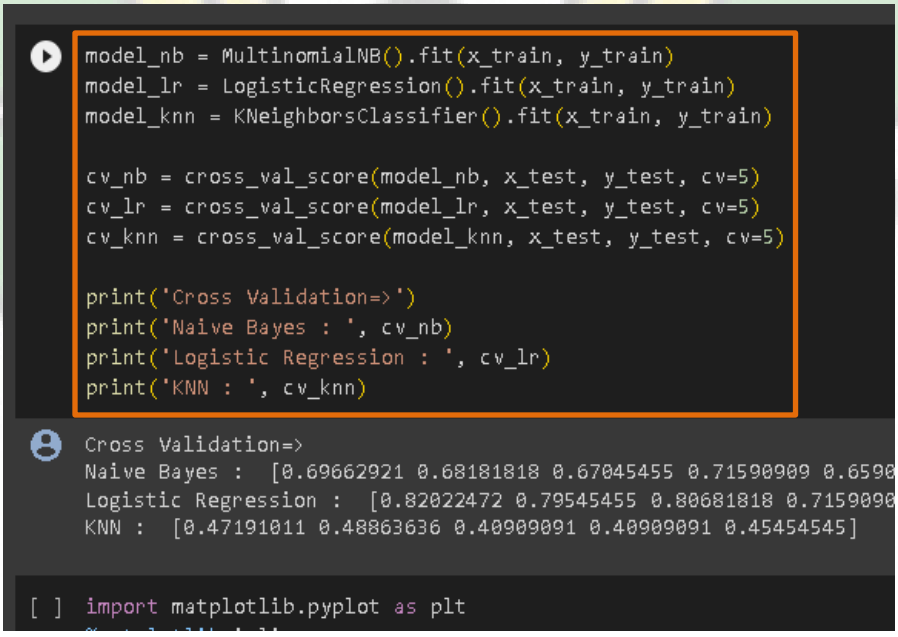
b. *K-Fold Cross Validation*

Fase selanjutnya yaitu penentuan *Hyperparameter* yang digunakan sebagai pelatihan model untuk memperkirakan hasil prediksi parameter model dengan menggunakan metode *K-fold cross validation* implementasi dari *library sklearn model_selection*.



Gambar 4.9. *Library K-fold cross validation*

Tujuan *K-Fold cross validation* tersebut yaitu untuk memperkirakan hasil probalitas algoritma *naïve bayes, logistic regression*, dan KNN agar terhindar dari *Overfitting* (kondisi dimana nilai akurasi pada data training yang tinggi jika dibandingkan dengan nilai akurasi data testing yang rendah).



Gambar 4.10. *K-Fold cross validation*

Pada penelitian ini, dataset yang terdiri dari data x (data vektor) dan data y (data sentimen analisis) akan dilakukan integrasi pengujian akurasi sebanyak 5-*Fold* (5 kali pengujian). Hasil *Hyperparameter* tersebut dapat di lihat pada tabel 4.2., tabel 4.3., dan tabel 4.4.

Tabel 4.2. *K-Fold cross naïve bayes*

K-Fold Cross Naïve Bayes			
No	**Fold**	**Akurasi**	**Nilai Rata - rata**
1	1	0.69%	0.68%
2	2	0.68%	
3	3	0.67%	
4	4	0.71%	
5	5	0.65%	

Berdasarkan pada tabel 4.2. menjelaskan, penggunaan 5-*Fold cross validation* pada algoritma *naïve bayes* memiliki nilai akurasi tertinggi sebesar 0.71% pada iterasi ke-4, sedangkan untuk nilai akurasi terendah terdapat pada iterasi ke-5 dengan nilai akurasi sebesar 0,65%, dan untuk nilai rata-rata dari keseluruhan akurasi pada *K-Fold Cross Naïve Bayes* sebesar 0.68%.

Tabel 4.3. *K-Fold Cross Logistic Regression*

K-Fold Cross Logistic Regression			
No	**Fold**	**Akurasi**	**Nilai Rata - rata**
1	1	0.82%	0.78%
2	2	0.79%	
3	3	0.80%	
4	4	0.71%	
5	5	0.78%	

Selanjutnya pada tabel 4.3. menjelaskan, penggunaan 5-*Fold cross validation* pada algoritma *logistic regression* memiliki nilai akurasi tertinggi sebesar 0.82% pada iterasi ke-1, sedangkan untuk nilai akurasi terendah terdapat pada iterasi ke-4 dengan nilai akurasi sebesar 0,71%, dan untuk nilai rata-rata dari keseluruhan akurasi pada *K-Fold Cross logistic regression* sebesar 0.78%.

Tabel 4.4. *K-Fold Cross* KNN

***K-Fold Cross* KNN**			
No	**Fold**	**Akurasi**	**Nilai Rata - rata**
1	1	0.47%	0.44%
2	2	0.48%	
3	3	0.40%	
4	4	0.40%	
5	5	0.45%	

Dan yang terakhir pada tabel 4.4. menjelaskan, penggunaan 5-*Fold cross validation* pada algoritma KNN memiliki nilai akurasi tertinggi sebesar 0.48% pada iterasi ke-2, sedangkan untuk nilai akurasi terendah terdapat pada iterasi ke-4 dan ke-5 dengan nilai akurasi sebesar 0,40%, dan untuk nilai rata-rata dari keseluruhan akurasi pada *K-Fold Cross* KNN sebesar 0.44%.

4.3. Evaluasi Model

Pada evaluasi model tersebut meliputi ke dalam 2 fase yaitu, fase klasifikasi dalam penentuan prediksi nilai kebenaran (*Confusion matrics*) dan fase pengujian pengukuran *performance* pada algoritma *naïve bayes, logistik regression,* dan KNN.

a. *Confusion Matrics*

Pada fase ini, model algoritma *naïve bayes, logistic regression* dan KNN akan dilakukan ke tahap klasifikasi model algoritma dengan menggunakan metode *convusion matrics* implementasi dari *library sklearn_ metrics.*

```
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

cm = confusion_matrix(y_test, y_pred)
cm

array([[ 16,  21,   4],
       [  1,  98,   2],
       [  4,  95, 200]])
```

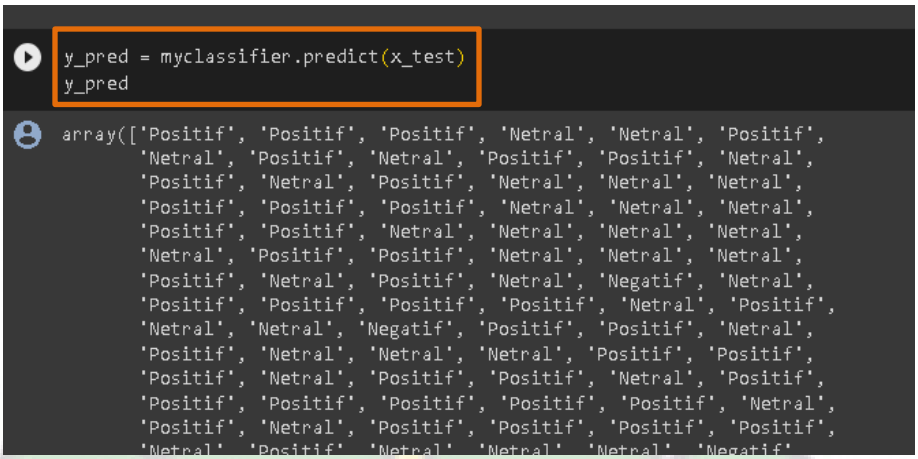
Gambar 4.11. *Library sklearn_metrics*

Sebelum dilakukan ke tahap klasifikasi, hasil dari penerapan modeling data tersebut akan dilakukan ke tahap pelatihan model algoritma dengan menggunakan **data *training* x** (data latih vektor) dan **data *training* y** (data latih sentimen analisis) sebagai pelatihan model algoritma.

```
[ ] myclassifier = KNeighborsClassifier(n_neighbors=1,weights='uniform',p=2)
    myclassifier.fit(x_train,y_train)

    KNeighborsClassifier
    KNeighborsClassifier(n_neighbors=1)
```

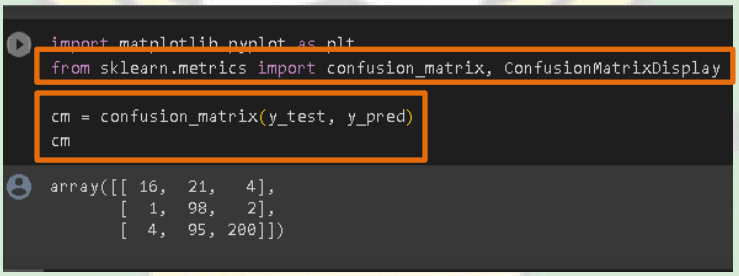
Gambar 4.12. Pelatihan model algoritma KNN

hasil dari pelatihan model algoritma tersebut akan dilakukan ke tahap pengujian model algoritma dengan menggunakan **data *testing* x** (data uji vektor) sebagai penguji model algoritma dan hasil dari pengujian model algoritma tersebut akan inputkan ke dalam **data y_pred** (data hasil pengujian algoritma).



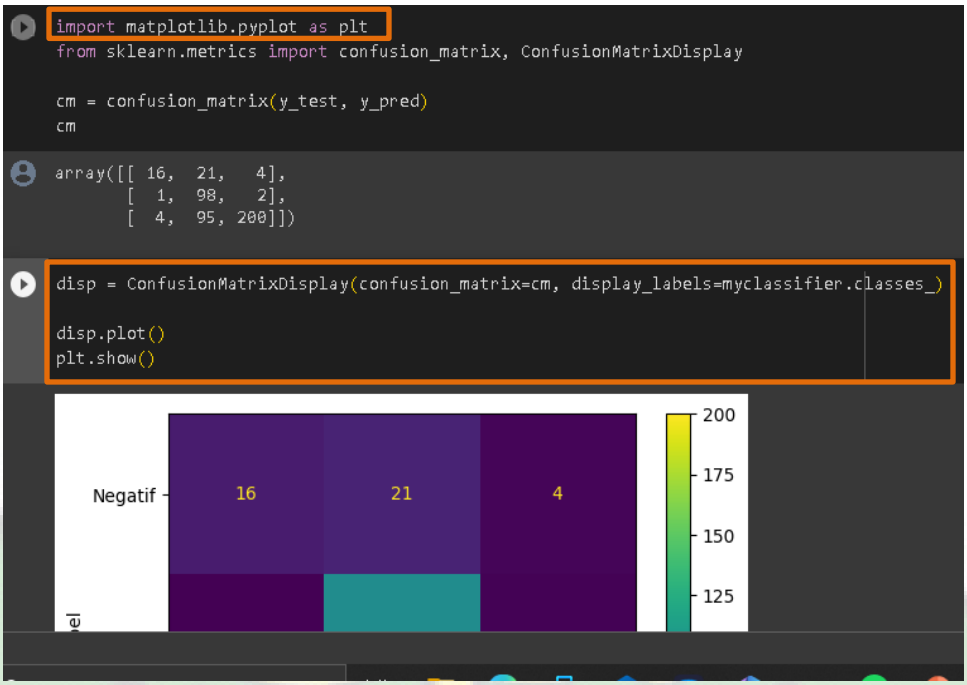
Gambar 4.13. Data y_pred

Setelah dilakukan pengujian model algoritma tersebut, akan di lakukan ke tahap perbandingan data antara **data *testing* y** (data uji sentimen analisis) dengan **data y_pred** dan hasil perbandingan data tersebut akan dilakukan ke tahap kesamaan data kedalam *library sklern_ metrics* sebagai hasil klasifikasi model algoritma.



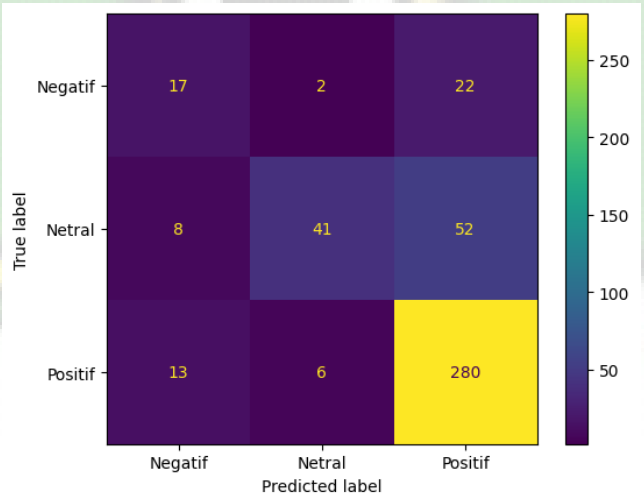
Gambar 4.14. Tahap kesamaan data

Hasil klasifikasi model algoritma tersebut akan dilakukan presentasi klasifikasi algoritma *naïve bayes, logistic regression*, dan KNN dalam bentuk *Matplotlip* implementasi dari *library matplotlib_pyplot*. *Matplotlip* merupakan sebuah *open-source library* yang bertujuan untuk memvisualisasi data dalam bentuk gambar untuk menampilkan suatu informasi.



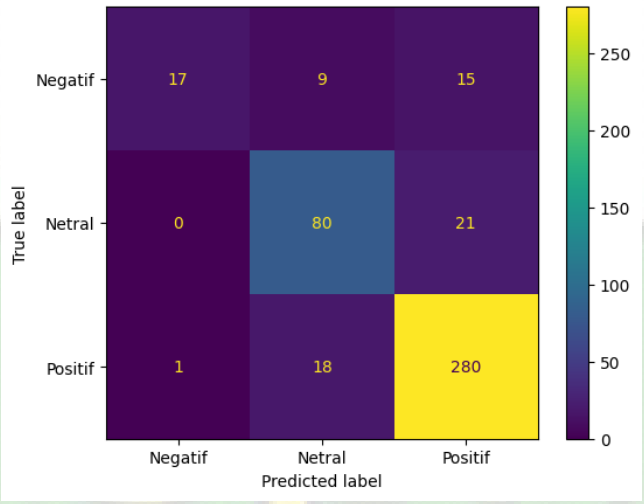
Gambar 4.15. *Matplolip* klasifikasi algoritma

Hasil evaluasi *convusion matrics* dapat di lihat pada gambar 4.5., gambar 4.6., dan gambar 4.7.



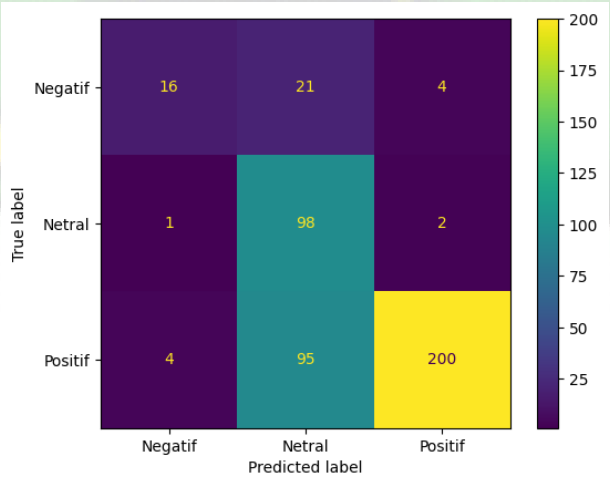
Gambar 4.16. *Confusion matrics* algoritma *naïve bayes*

Pada gambar 4.5 menjelaskan, *confusion matrics* algoritma *naïve bayes* pada data sentimen analisis terdapat 280 dataset yang dinyatakan benar – benar positif, 17 dataset yang dinyatakan benar - benar negatif, dan 41 dataset dinyatakan benar – benar netral.



Gambar 4.17. *Confusion matrics* algoritma *logistic regression*

Selanjutnya pada gambar 4.6. menjelaskan, *confusion matrics* algoritma *logistic regression* pada data sentimen analisis terdapat 280 dataset yang dinyatakan benar – benar positif, 17 dataset yang dinyatakan benar - benar negatif, dan 80 dataset dinyatakan benar – benar netral.



Gambar 4.18. *Confusion matrics* pada algoritma KNN

Dan yang terakhir pada gambar 4.7. menjelaskan, *confusion matrics* algoritma KNN pada data sentimen analisis terdapat 200 dataset yang dinyatakan benar – benar positif, 16 dataset yang dinyatakan benar – benar negatif, dan 98 dataset dinyatakan benar – benar netral.

b. Pengukuran *Performance Vector*

Pada fase ini, data yang sudah di klasifikasi ke dalam algoritma *Naïve bayes, Logistic regression,* dan KNN akan dilakukan perhitungan *Precision, Recall, F1-score, Macro avg,* dan *Weighted avg* dengan mengimplementasikan data kedalam *library sklearn_metrics* untuk mendapatkan hasil nilai akurasi dari pengujian antara data yang di latih ke dalam pemodelan algoritma dengan data asli (data sentimen analisis).

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

     Negatif       0.76      0.39      0.52        41
      Netral       0.46      0.97      0.62       101
     Positif       0.97      0.67      0.79       299

    accuracy                           0.71       441
   macro avg       0.73      0.68      0.64       441
weighted avg       0.83      0.71      0.73       441
```

Gambar 4.19. *Performance Vector* pada algoritma KNN

Keterangan pada gambar 4. dari hasil pengujian *Performance Vector* menjelaskan sebagai berikut:

- *Precision* merupakan nilai kecocokan data antara data yang dimodelkan kedalam algoritma dengan data asli (data sentimen analisis) dalam pengklasifikasian [5].
- *Recall* merupakan nilai prediksi dalam pernyataan benar antara data yang dimodelkan kedalam algoritma dengan data asli [6].

- *F1-score* merupakan nilai rata – rata antara *Precision* dan *Recall* yang dibobotkan [7].
- *Support* merupakan jumlah data yang di keluarkan
- *Macro avg* merupakan perhintungan metrik (seperti akurasi, *precision, recall,* dll.) secara independen untuk setiap kelas, lalu dari hasil metrik tersebut akan dilakukan nilai rata – rata. [8].
- *Weighted avg* merupakan perhintungan metrik untuk setiap kelas, lalu hasil metrik tersebut dikalikan dengan bobot yang terdapat pada kelas tersebut sebelum dijumlahkan dari hasil keseluruhan metrik. [9].

Berikut hasil pengujian *Performance vector* pada kelas sentimen analisis dapat dilihat pada tabel 4.5 sebagai berikut.

Tabel 4.5. pengujian *Performance vector* kelas sentimen analisis

No	Algoritma	Analisis Sentimen	*Performance vector*			
			Precision	*Recall*	*F1-score*	*Accuracy*
1	*Naïve bayes*	Positif	0.79%	0.94%	0.86%	0.77%
		Negatif	0.45%	0.41%	0.43%	
		Netral	0.84%	0.41%	0.55%	
2	*Logistic regression*	Positif	0.89%	0.94%	0.91%	0.85%
		Negatif	0.94%	0.41%	0.58%	
		Netral	0.75%	0.79%	0.77%	
3	KNN	Positif	0.97%	0.67%	0.79%	0.71%
		Negatif	0.76%	0.39%	0.52%	
		Netral	0.46%	0.97%	0.62%	

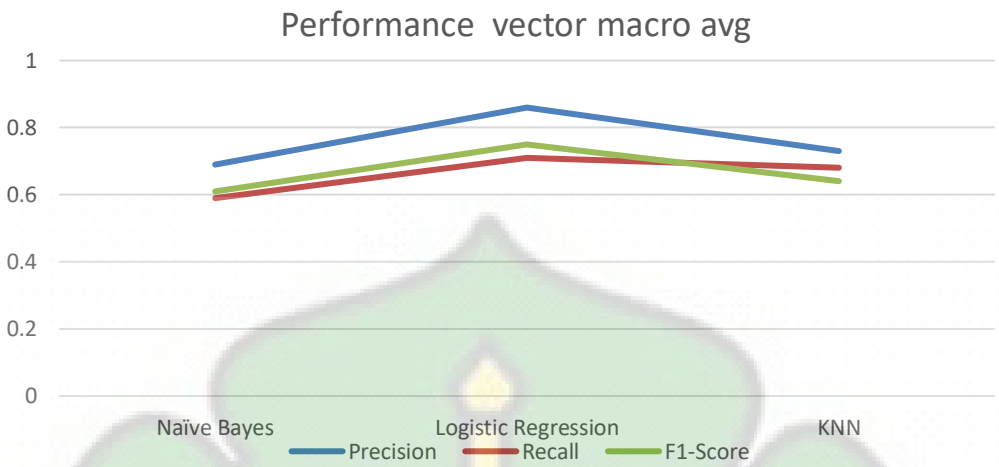
Berdasarkan hasil pengujian *Performance vector* pada kelas sentimen analisis menjelaskan, perolehan nilai terbaik pada algoritma *Naïve bayes* untuk skor hasil

pengujian tertinggi terdapat pada pengujian *Recall* data positif sebesar 0.94%, sedangkan untuk skor hasil pengujian terendah terdapat pada pengujian *Recall* data negatif dan data netral sebesar 0.41%, dan untuk perolehan nilai rata – rata tertinggi terdapat pada data positif sebesar 0.86% dengan hasil akurasi sebesar 0.77%. Selanjutnya perolehan nilai terbaik pada algoritma *Logistic regression* untuk skor hasil pengujian tertinggi terdapat pada pengujian *Recall* data positif dan pengujian *Precision* data negatif sebesar 0.94%, sedangkan untuk skor hasil pengujian terendah terdapat pada pengujian *Recall* data negatif sebesar 0.41%, dan untuk perolehan nilai rata – rata tertinggi terdapat pada data positif sebesar 0.91% dengan hasil akurasi sebesar 0.85%. Dan yang terakhir perolehan nilai terbaik pada algoritma KNN untuk skor hasil pengujian tertinggi terdapat pada pengujian *Precision* data positif dan pengujian *Recall* data netral sebesar 0.97%, sedangkan untuk skor hasil pengujian terendah terdapat pada pengujian *Recall* data negatif sebesar 0.39%, dan untuk perolehan nilai rata – rata tertinggi terdapat pada data positif sebesar 0.79% dengan hasil akurasi sebesar 0.71%.

Selanjutnya perhintungan *Macro avg* untuk nilai rata - rata keseluruhan data dapat dilihat pada tabel 4.6 dan gambar 4.9.

Tabel 4.6. *Macro avg* untuk nilai rata – rata keseluruhan

No	Algoritma	*Macro avg*		
		Precision	*Recall*	*F1-score*
1	*Naïve bayes*	0.69%	0.59%	0.61%
2	*Logistic regression*	0.86%	0.71%	0.75%
3	KNN	0.73%	0.68%	0.64%



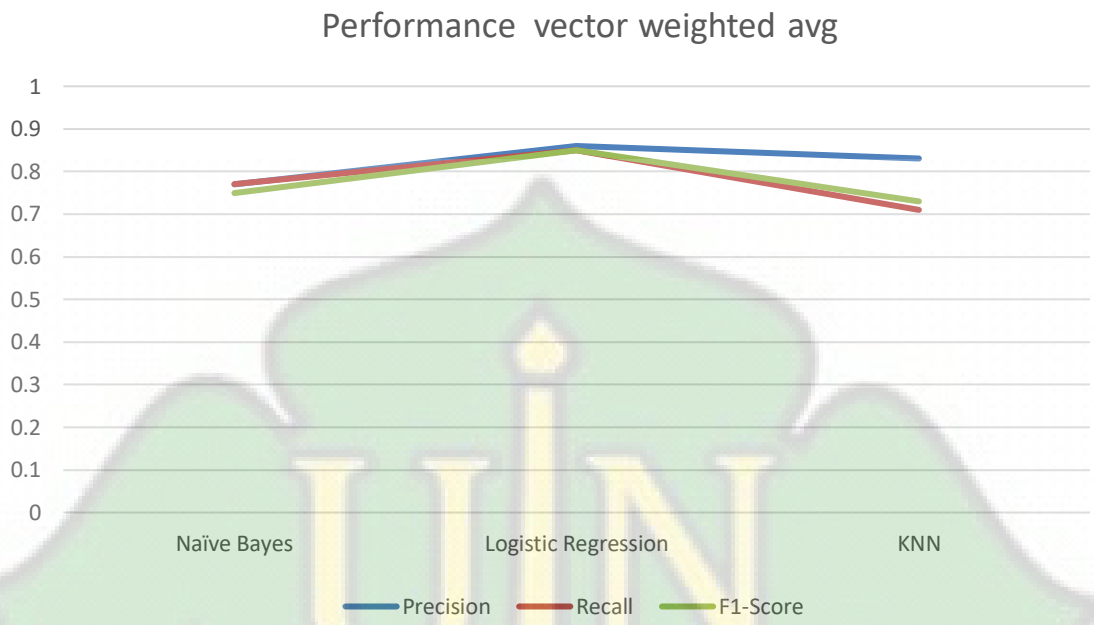
Gambar 4.20. Diagram garis *Macro avg*

Perbandingan hasil nilai rata - rata keseluruhan data atau *Macro avg* menjelaskan, perolehan nilai tertinggi pada *Precision* tersebut terdapat pada algoritma *Logistic regression* dengan skor 0.86%, sedangkan untuk nilai terendah pada *Precision* terdapat pada algoritma *Naïve bayes* dengan skor 0.69%. Selanjutnya perolehan nilai tertinggi pada *Recall* tersebut terdapat pada algoritma *Logistic regression* dengan skor 0.71%, sedangkan untuk nilai terendah pada *Recall* terdapat pada algoritma *Naïve bayes* dengan skor 0.59%. dan untuk nilai rata – rata atau *F1-score* tertinggi terdapat pada algoritma *Logistic regression* dengan skor 0.75%, sedangkan untuk nilai rata – rata terendah terdapat pada algoritma *Naïve bayes* dengan skor 0.61%.

Selanjutnya untuk nilai rata – rata tertimbang dari keseluruhan data atau *Weighted avg* dapat dilihat pada tabel 4.7 dan gambar 4.10.

Tabel 4.7. Hasil *Weighted avg*

No	Algoritma	*Weighted avg*		
		Precision	*Recall*	*F1-score*
1	*Naïve bayes*	0.77%	0.77%	0.75%
2	*Logistic regression*	0.86%	0.85%	0.85%
3	KNN	0.83%	0.71%	0.73%



Gambar 4.21. Diagram garis *Weighted avg*

Perbandingan hasil nilai rata – rata tertimbang atau *weighted avg* menjelaskan, perolehan nilai tertinggi pada *Precision* tersebut terdapat pada algoritma *Logistic regression* dengan skor 0.86%, sedangkan untuk nilai terendah pada *Precision* terdapat pada algoritma *Naïve bayes* dengan skor 0.77%. Selanjutnya perolehan nilai tertinggi pada *Recall* tersebut terdapat pada algoritma *Logistic regression* dengan skor 0.85%, sedangkan untuk nilai terendah pada *Recall* terdapat pada algoritma KNN dengan skor 0.71%. Dan untuk nilai rata – rata atau *F1-score* tertinggi terdapat pada algoritma *Logistic regression* dengan skor 0.85%, sedangkan untuk nilai rata – rata terendah terdapat pada algoritma KNN dengan skor 0.73%.

Berdasarkan pemilihan hasil evaluasi model antara *Macro avg* dan *Weighted avg* dapat dijelaskan sebagai berikut:

1. jika untuk mengetahui kinerja model secara keseluruhan tanpa mengetahui seberapa besar atau kecilnya kelas, mungkin akan menggunakan *Macro avg* sebagai hasil evaluasi model.

2. dan apabila misalnya, jika terdapat dua kelas yang terdiri kelas negatif dengan 90 *Instance* (*Instance* bisa di artikan sebagai wujud kelas. Contoh, sebuah rumah di dalamnya ada tv, kursi, sofa, dan alat – alat lainnya) dan kelas positif dengan 10 *Instace*, namun kelas positif adalah kelas yang lebih penting dan ingin memastikan bahwa kinerja pada kelas positif diberi bobot yang lebih besar, maka akan menggunakan *Weighted avg* sebagai hasil evaluasi model.

Pemilihan hasil evaluasi model antara *Macro avg* dan *Weighted avg* tergantung pada konteks spesifik dari masalah klasifikasi dan seberapa pentingnya masing – masing kelas dalam performa model. Oleh karena itu, untuk hasil evaluasi model ini akan mengambil *Weihgted avg* sebagai hasil evaluasi model.

4.4. Visualisasi Hasil

Visualisasi hasil bertujuan untuk mempresentasikan hasil dari suatu kelompok data yang menampilkan kata yang sering muncul yang ada pada opini data sentimen positif dan negatif dengan menggunakan metode *Word Cloud* implementasi dari *library wordcloud*. Hasil visualisasi data sentimen positif dan negatif dapat di lihat pada gambar 4.12 dan gambar 4.13

Gambar 4.22. *Word Cloud* sentimen positif

Hasil visualisasi data sentimen positif pada gambar 4.12 menjelaskan, kata – kata yang sering muncul pada data sentimen analisis positif terdiri dari kata “permaisuri”, “nyonya”, “peluang”, “perjuangan”, “berkah”, “senior”, “uph”, “artai”, “itu”, “sosok”, “di”, “ppp”, “sebagai”, “sementara”, “ahli”, “politik”, “pantas”, “dpp”, “besar”, “ketua”, “mente”, “xixixi”, “tidak”, “komunikasi”, “melirik”, “kinerja”, “pdi” “tweet”, “posisi”, “puan”, “dipastikan”, dan “golkar”.

Gambar 4.23. *Word Cloud* sentimen negatif

Hasil visualisasi data sentimen negatif pada gambar 4.13 menjelaskan, kata – kata yang sering muncul pada data sentimen analisis negatif terdiri dari kata “nyonya”, “benar”, “satu”, “tidak”, “lagi”, “langkah”, “kata”, “da”, “leg”, “dihina”, “permaisuri”, “gaberani”, “pada”, “curse”, “tweet”, “kamu”, “gabung”, “jangan”, “ini”, “berkata”, “kecemasan”, “wartawan”, “ini”, “sukarelawan”, “benar”, “pmp”, “ya”, “mala”, “pak”, “malam”, “mah”, dan “puan”.

BAB V
KESIMPULAN DAN SARAN

1. Kesimpulan

Berdasarkan hasil pembahasan pada bab sebelumnya dari hasil penelitian ini dapat disimpulkan sebagai berikut:

a. Hasil sentimen analisis pada dataset dari hasil *crawling* data pada twitter dengan kata kunci "calon presiden" yang di ambil dari tanggal 18 Januari 2023 sampai 25 Januari 2023 dengan jumlah 2201 dataset yang terkumpul menjelaskan, terdapat 1471 dataset mengandung positif artinya kecenderungan masyarakat terdapat pujian atau dukungan terhadap tokoh Bacapres untuk maju Capres 2024. Sedangkan 197 dataset mengandung negatif menjelaskan kecenderungan masyarakat terdapat ketidakpuasan terhadap tokoh Bacapres untuk maju Capres 2024. dan yang terakhir 533 dataset mengandung netral menjelaskan kecenderungan masyarakat terdapat tidak keterlibatan dalam mendukung bacapres untuk maju Capres 2024.

b. Pengujian *K-fold cross validation* dengan melakukan pengujian 5 kali (*fold*) pada data sentimen analisis terdapat perbedaan nilai akurasi dari masing – masing algoritma tersebut, dimana pada algoritma *Naïve bayes* menghasilkan nilai rata – rata akurasi 0.68% dengan nilai akurasi tertinggi sebesar 0.71% pada iterasi ke-4 lebih rendah dari hasil akurasi algortima *Logistic regression* dengan perolehan nilai rata – rata akurasi 0.78% dengan nilai akurasi tertinggi sebesar 0.82% pada iterasi ke-1. Dan yang terakhir pada algoritma KNN menghasilkan nilai rata – rata akurasi 0,44% dengan akurasi tertinggi sebesar 0.48% pada iterasi ke-2 lebih rendah dari nilai akurasi algoritma *Naïve bayes* dan *Logistic regression*.

c. Evaluasi model pada data sentimen analisis menjelaskan:
   - Hasil evaluasi algoritma *Naïve bayes* pada data sentimen analisis menghasilkan nilai akurasi sebesar 0.77%, dengan nilai rata - rata *Precision* keseluruhan sebesar 0.77%, nilai rata – rata *Recall* keseluruhan sebesar 0.77%, dan nilai rata – rata atau *F1-score* keseluruhan sebesar 0.75%.

- Hasil evaluasi algoritma *Logistic regression* pada data sentimen analisis menghasilkan nilai akurasi sebesar 0.85%, dengan nilai rata – rata *Precision* keseluruhan sebesar 0.86%, nilai rata - rata *Recall* keseluruhan sebesar 0.85%, dan nilai rata – rata atau *F1-score* keseluruhan sebesar 0.85%.
- Dan yang terakhir hasil evaluasi algoritma KNN pada data sentimen analisis menghasilkan nilai akurasi sebesar 0.71%, dengan nilai rata - rata *Precision* keseluruhan sebesar 0.83%, nilai rata – rata *Recall* keseluruhan sebesar 0.71%, dan nilai rata – rata atau *F1-score* keseluruhan sebesar 0.73%.

2. Saran

Berdasarkan pada penelitian menjelaskan, pengujian klasifikasi data sentimen analisis dengan algoritma *Logistic regression* menghasilkan akurasi tinggi jika dibandingkan dengan algoritma *Naïve bayes* dan KNN sehingga pada penelitian selanjutnya di perlukan metode algoritma lainnya dalam pengujian klasifikasi pada data sentimen analisis untuk di jadikan perbandingan pada penelitian sebelumnya.

DAFTAR PUSTAKA

Akmal, A. D., Permana, I., Fajri, H., & Yuliarti, Y. (2022). Opini Masyarakat Twitter terhadap Kandidat Bakal Calon Presiden Republik Indonesia Tahun 2024. *Jurnal Manajemen dan Ilmu Administrasi Publik (JMIAP)*, *4*(4), 292–300. https://doi.org/10.24036/jmiap.v4i4.160

Apriani, R., Gustian, D., Program, S., Sistem, I., Putra, U. N., Indonesia, S., Raya, J., Kaler, C., 21, N., & Sukabumi, K. (2019). ANALISIS SENTIMEN DENGAN NAÏVE BAYES TERHADAP KOMENTAR APLIKASI TOKOPEDIA. Dalam *Jurnal Rekayasa Teknologi Nusa Putra* (Vol. 6, Nomor 1).

Assaidi, S. A., & Amin, F. (t.t.). *Analisis Sentimen Evaluasi Pembelajaran Tatap Muka 100 Persen pada Pengguna Twitter menggunakan Metode Logistic Regression*.

Bahtiar, S. A. H., Dewa, C. K., & Luthfi, A. (2023). Comparison of Naïve Bayes and Logistic Regression in Sentiment Analysis on Marketplace Reviews Using Rating-Based Labeling. *Journal of Information Systems and Informatics*, *5*(3), 915–927. https://doi.org/10.51519/journalisi.v5i3.539

Fais Sya' bani, M. R., Enri, U., & Padilah, T. N. (2022). Analisis Sentimen Terhadap Bakal Calon Presiden 2024 Dengan Algoritme Naïve Bayes. *JURIKOM (Jurnal Riset Komputer)*, *9*(2), 265. https://doi.org/10.30865/jurikom.v9i2.3989

Fikri, M. I., Sabrila, T. S., & Azhar, Y. (2020). Perbandingan Metode Naïve Bayes dan Support Vector Machine pada Analisis Sentimen Twitter. *SMATIKA JURNAL*, *10*(02), 71–76. https://doi.org/10.32664/smatika.v10i02.455

Halim, E. N., Huda, B., & Elanda, A. (t.t.-a). Jawa Barat 41361 3 Program Studi Teknik Informatika. Dalam *STMIK Rosma Jl. Parahiyangan, Adiarsa Bar., Kec. Karawang Bar*. www.jurnal.unimed.ac.id

Halim, E. N., Huda, B., & Elanda, A. (t.t.-b). Jawa Barat 41361 3 Program Studi Teknik Informatika. Dalam *STMIK Rosma Jl. Parahiyangan, Adiarsa Bar., Kec. Karawang Bar*. www.jurnal.unimed.ac.id

Hasibuan, E., & Heriyanto, E. A. (t.t.). ANALISIS SENTIMEN PADA ULASAN APLIKASI AMAZON SHOPPING DI GOOGLE PLAY STORE MENGGUNAKAN NAIVE BAYES CLASSIFIER. *JTS*, *1*(3).

Keputusan Direktur Jenderal Pendidikan Tinggi, S., dan Teknologi Nomor, R., Putri Nardilasari, A., Lia Hananto, A., Shofia Hilabi, S., Priyatna, B., Studi Sistem Informasi, P., Ilmu Komputer, F., & Buana Perjuangan Karawang, U. (2026). *Terakreditasi SINTA Peringkat 3 Analisis Sentimen Calon Presiden 2024 Menggunakan Algoritma SVM Pada Media Sosial Twitter* (Vol. 7, Nomor 1).

KOMPARASI ALGORITMA SUPPORT VECTOR MACHINE DAN K-NEAREST NEIGHBOR BERBASIS PARTICLE SWARM OPTIMIZATION PADA ANALISIS SENTIMEN FENOMENA TAGAR #2019GANTIPRESIDEN. (t.t.).

Manalu, D. R., L. Tobing, M. C., & Yohanna, M. (2022). ANALISIS SENTIMEN TWITTER TERHADAP WACANA PENUNDAAN PEMILU DENGAN METODE SUPPORT VECTOR MACHINE. *METHOMIKA Jurnal Manajemen Informatika dan Komputerisasi Akuntansi*, *6*(6), 149–156. https://doi.org/10.46880/jmika.Vol6No2.pp149-156

Oleh. (t.t.). *ANALYSIS SENTIMEN PADA MEDIA SOSIAL MENGGUNAKAN METODE NAIVE BAYES CLASSIFIER TESIS*.

Perdana, A., Hermawan, A., & Avianto, D. (2022). Analisis Sentimen Terhadap Isu Penundaan Pemilu di Twitter Menggunakan Naive Bayes Clasifier. *Jurnal Sisfokom (Sistem Informasi dan Komputer)*, *11*(2), 195–200. https://doi.org/10.32736/sisfokom.v11i2.1412

Praise for Natural Language Processing with Transformers. (t.t.).

Rahayu, S., Asmoro, B. R., & Rinal, E. (2023). Classification of Congestion in Jakarta Using KNN, Naïve Bayes and Decision Tree Method. *Jurnal Syntax Admiration*, *4*(7), 928–952. https://doi.org/10.46799/jsa.v4i7.654

Rahman, A., #1, I., Sulistiani, H., Miftaq, B., #3, H., Nurkholis, A., & #5, S. (t.t.). *JEPIN (Jurnal Edukasi dan Penelitian Informatika) Analisis Perbandingan Algoritma LSTM dan Naive Bayes untuk Analisis Sentimen*.

Ramadhani, S. H., & Wahyudin, M. I. (2022). Analisis Sentimen Terhadap Vaksinasi Astra Zeneca pada Twitter Menggunakan Metode Naïve Bayes dan K-NN. *Jurnal Teknologi Informasi dan Komunikasi)*, *6*(4), 2022. https://doi.org/10.35870/jti

Thet, T. T., Na, J. C., & Khoo, C. S. G. (2010). Aspect-based sentiment analysis of movie reviews on discussion boards. *Journal of Information Science*, *36*(6), 823–848. https://doi.org/10.1177/0165551510388123

LAMPIRAN

Lampiran 1: *Crawling* data (pengumpulan data)

```
!git clone --depth=1 https://github.com/twintproject/twint.git
%cd twint
!pip3 install . -r requirements.txt
```

- **"Git clone" merupakan tindakan untuk menyalin dari *repository* online ke *repository* lokal (mendownload *repository* ke dalam komputer)**
- **"--depth=1 https://github.com/twintproject/twint.git" merupakan kata kunci untuk penginstalan *library twint* sebagai pengambil data twitter tanpa menggunakan API twitter**
- **"pip3 install . -r requirements.txt" merupakan penginstalan paket python berfungsi sebagai dokumentasi dan alat pengendali versi untuk package yang akan digunakan.**

```
!pip install aiohttp==3.7.0
```

- **"pip install aiohttp==3.7.0" merupakan penginstalan paket *aiohttp* (client/server http untuk asyncio) berfungsi untuk membuat aplikasi dan paket asinkron (komunikasi yang dilakukan secara tertunda atau tidak langsung) lebih mudah**

```
!pip install nest-asyncio
import nest_asyncio
nest_asyncio.apply()
```

- **"!pip install nest-asyncio import nest_asyncio" merupakan penginstalan paket *asyncio* berfungsi untuk berisikan perintah yang ingin diproses.**

```
import nest_asyncio
```

```
nest_asyncio.apply() #digunakan sekali untuk mengaktifkan tindakan
serentak dalam notebook jupyter.
import twint #untuk import twint


c = twint.Config()
c.Search = 'calon presiden'
c.Pandas = True
twint.run.Search(c)
```

- ***Code*** **diatas merupakan pengambilan data pada twitter dengan kata kunci "calon presiden"**

```
Tweets_df = twint.storage.panda.Tweets_df
Tweets_df.head()


Tweets_df.to_csv("calon_presiden.csv", index=False)
```

- ***Code*** **diatas merupakan penginputan data ke dalam csv**

Lampiran 2: *Preprocesing* (pembersihan data)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import re, string
import nltk
import openpyxl


!pip install Sastrawi
```

- **"`!pip install Sastrawi`" merupakan penginstalan paket *Sastrawi* berfungsi menyesuaikan kata teks dengan standar kamus bahasa indonesia**

```
!pip install StemmerFactory

!pip install Swifter


from nltk.tokenize import word_tokenize
```

```
from nltk.corpus import stopwords
from nltk.stem import SnowballStemmer
from nltk.corpus import wordnet
from nltk.stem import WordNetLemmatizer
from nltk.probability import FreqDist
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')
nltk.download('omw-1.4')
```

- ***Code*** **diatas merupakan penginstalan paket** ***Preprocessing***

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import swifter


from google.colab import files
uploades = files.upload()


df_penerima = pd.read_csv ('calon_presiden.csv')
df_penerima.head()
```

- ***Code*** **diatas merupakan penginputan data untuk di lakukan** ***Preprocessing***

```
df_penerima['tweet'] = df_penerima['tweet'].str.lower()

print('Case Folding Result : \n')
print(df_penerima['tweet'].head(5))
print('\n\n\n')
```

- ***Code*** **diatas merupakan pengambilan data yang terdapat pada kolom "tweet"**

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

```
  text = ' '.join(re.sub("([@#][A-Aa-z0-9]+)|(\w+:\/\/\S+)"," ",
text).split())
  return text.replace("http://", " ").replace("https://", " ")
df_penerima['tweet'] =
df_penerima['tweet'].apply(remove_tweet_special)

def remove_number(text):
  return re.sub(r"\d", "", text)
df_penerima['tweet'] = df_penerima['tweet'].apply(remove_number)

def remove_punctuation(text):
  return text.translate(str.maketrans("","",string.punctuation))
df_penerima['tweet'] =
df_penerima['tweet'].apply(remove_punctuation)

def remove_whitespace_LT(text):
  return text.strip()
df_penerima['tweet'] =
df_penerima['tweet'].apply(remove_whitespace_LT)

def remove_whitespace_multiple(text):
  return re.sub('\s',' ',text)
df_penerima['tweet'] =
df_penerima['tweet'].apply(remove_whitespace_multiple)

def remove_singl_char(text):
  return re.sub(r"\b[a-zA-Z]\b","",text)
df_penerima['tweet'] =
df_penerima['tweet'].apply(remove_singl_char)
```

- ***Code*** **diatas merupakan tahap** ***Cleansing***

```
def word_tokenize_wrapper(text):
  return word_tokenize(text)
df_penerima['tweet'] =
df_penerima['tweet'].apply(word_tokenize_wrapper)


print('Tokenizing Result : \n')
print(df_penerima['tweet'].head(10))
print('\n\n\n')
```

```
def FreqDist_wrapper(text):
  return FreqDist(text)
df_penerima['tweet_token_fdist'] =
df_penerima['tweet'].apply(FreqDist_wrapper)

print('Frequency Tokens : \n')
print(df_penerima['tweet_token_fdist'].head(10).apply(lambda x :
x.most_common()))
```

- ***Code* diatas merupakan tahap *Tokenizing***

```
list_stopwords = stopwords.words('indonesian')
list_stopwords.extend(['yg','dg','rt','dgn','ny','d','klo',
                       'kalo','amp','blar','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','xixixi','ucl'])


txt_stopword = pd.read_csv("calon_presiden.csv", names =
["stopwords"], header =None)
list_stopwords.extend(txt_stopword["stopwords"][0].split(' '))
list_stopwords = set(list_stopwords)

def stopwords_removal(words):
  return [word for word in words if word not in list_stopwords]
df_penerima['tweet_tokens_wsw'] =
df_penerima['tweet'].apply(stopwords_removal)
```

- ***Code* diatas merupakan tahap *Stopword Removal***

```
print(df_penerima['tweet_tokens_wsw'].head(10))


normalizad_word = pd.read_csv('calon_presiden.csv')

normalizad_word_dict = {}

for index, row in normalizad_word.iterrows():
```

```
  if row[0] not in normalizad_word.iterrows():
    normalizad_word_dict[row[0]] = row[1]

def normalizad_term(document):
  return [normalizad_word_dict[term] if term in
normalizad_word_dict else term for term in document]
df_penerima['tweet_normalized'] =
df_penerima['tweet_tokens_wsw'].apply(normalizad_term)


df_penerima['tweet_normalized'].head(10)


factory = StemmerFactory()
stemmer = factory.create_stemmer()

def stemmed_wrapper(term):
  return stemmer.stem(term)

term_dict = {}

for document in df_penerima['tweet_normalized']:
  for term in document:
    if term not in term_dict:
      term_dict[term] = ' '

print(len(term_dict))
print("_ _ _ _ _ _ _ _ _ _ _ _ _ _")

def get_stemmed_term(document):
  return [term_dict[term] for term in document]
df_penerima['tweet_tokens_stemmed'] =
df_penerima['tweet_normalized'].swifter.apply(get_stemmed_term)

print(df_penerima['tweet_tokens_stemmed'])
```

- ***Code*** **diatas merupakan tahap** ***Normalized***

```
df_penerima.to_csv("preprocessing.csv")
```

Lampiran 3: Sentimen Analisis

```
!pip install tweet-preprocessor
!pip install textblob
!pip install sastrawi
!pip install emoji
!pip install pySastrawi


import os
import pandas as pd
import tweepy
import re
import string
from textblob import TextBlob
import preprocessor as p
from preprocessor.api import clean, tokenize, parse
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import datetime
from datetime import timedelta
import numpy as np
import emoji
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.feature_extraction.text import CountVectorizer
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
from Sastrawi.StopWordRemover.StopWordRemoverFactory import
StopWordRemoverFactory


import pandas as pd
def load_data():
  data = pd.read_csv('/content/tweet.csv')
  return data


tweet_df = load_data()
```

```
tweet_df = pd.DataFrame(tweet_df[['tweet']])
tweet_df.head


tweet_df.head(1000)


tweet_df = tweet_df[tweet_df['tweet']!='']
tweet_df.head(1000)


#reset index
tweet_df = tweet_df.reset_index(drop=True)
tweet_df.head()


!pip3 install googletrans==3.1.0a0


#translate
import pandas as pd
import googletrans
from googletrans import Translator


df = pd.read_csv('/content/tweet.csv')
df.head(1000)


translator =Translator()
translations = {}
for column in df.columns:
  unique_elements = df[column].unique()
  for element in unique_elements:
    translations[element] = translator.translate(element).text
translations
```

- ***Code*** **diatas merupakan** ***translate*** **kata dari bahasa indonesia ke bahasa inggris.**

```
df.replace(translations, inplace = True)
df.head(1000)
```

```
df.to_csv('/content/translate.csv',encoding='utf8', index=False)
```

```
import pandas as pd
def load_data():
  data = pd.read_csv('/content/translate.csv')
  return data
```

```
tweet_df = load_data()
tweet_df.head(1000)
```

```
#Lexicion Based
!pip install VaderSentiment
```

```
from vaderSentiment.vaderSentiment import
SentimentIntensityAnalyzer
analyser = SentimentIntensityAnalyzer()
```

- ***Code* diatas merupakan penginstallan *library VaderSentimen* berfungsi untuk menganalisis sentimen teks kalimat**

```
scores = [analyser.polarity_scores(x) for x in tweet_df['tweet']]
print(scores)
tweet_df['Compound_Score'] = [x['compound'] for x in scores]
```

```
tweet_df.head(1000)
```

```
tweet_df.nsmallest(1000, ['Compound_Score'])
```

```
#compound score lexicon based
tweet_df.loc[tweet_df['Compound_Score'] < 0, 'Sentiments'] =
'Negatif'

tweet_df.loc[tweet_df['Compound_Score'] == 0, 'Sentiments'] =
'Netral'
```

```
tweet_df.loc[tweet_df['Compound_Score'] > 0, 'Sentiments'] =
'Positif'

tweet_df.head(1000)

#compound score lexicon based
tweet_df.loc[tweet_df['Compound_Score'] < 0, 'Labels'] = '0'

tweet_df.loc[tweet_df['Compound_Score'] == 0, 'Labels'] = '1'

tweet_df.loc[tweet_df['Compound_Score'] > 0, 'Labels'] = '2'

tweet_df.head(1000)
```

- ***Code*** **diatas merupakan tahap pelabelan data sentimen analisis**

```
tweet_df.to_csv('/content/hasil_SentimenAnalisis.csv',encoding =
'utf8', index = False)
```

Lampiran 4: Penerapan modeling data dan pembagian data

```
import pandas as pd
import numpy as np
import string
import re
import nltk
nltk.download('punkt')
nltk.download('stopwords')
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.corpus import stopwords


!pip install tweet-preprocessor
!pip install emoji
!pip install sastrawi
!pip install pySastrawi
!pip install vaderSentiment
```

```
!pip install Sastrawi
!pip install swifter
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import swifter


from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
```

- “`from sklearn.feature_extraction.text import CountVectorizer`” **penginstalan *library* ini berfungsi untuk pengubahan kata teks tipe objek ke dalam matriks**
- “`from sklearn.model_selection import train_test_split`” **penginstalan *library* ini berfungsi untuk pembagian data ke dalam data *training* dan data *testing***
- “`from sklearn.naive_bayes import MultinomialNB`” **merupakan *library* algoritma *naïve bayes***
- “`from sklearn.linear_model import LogisticRegression`” **merupakan *library* algoritma *logistic regression***
- “`from sklearn.neighbors import KNeighborsClassifier`” **merupakan *library* algoritma KNN**

```
dataset = pd.read_csv("/content/hasil_SentimenAnalisis.csv")
dataset.head()


dataset.dtypes


dataset['Sentiments'].value_counts()


#vectorisasi data dan convert ke array
dataset['tweet'] = dataset['tweet'].astype(str)

vec = CountVectorizer().fit(dataset['tweet'])
vec_transform = vec.transform(dataset['tweet'])
```

```
print(vec_transform)
```

- ***Code*** **diatas merupakan tahap pengubahan kata teks tipe objek yang terdapat pada kolom "*tweet*" ke dalam matriks untuk dapat terbaca ke dalam algoritma**

```
dataset.shape


x = vec_transform.toarray()
y = dataset['Sentiments']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size
= 0.2)
```

- ***Code*** **diatas merupakan tahap pembagian data ke dalam data *training* dan data *testing***

```
x_train.shape


x_test.shape


y_train.shape


y_test.shape
```

Lampiran 5: Metode *cross validation*

```
from sklearn.model_selection import cross_val_score
```

- **"`from sklearn.model_selection import cross_val_score`" merupakan penginstalan *library cross validation***

```
model_nb = MultinomialNB().fit(x_train, y_train)
model_lr = LogisticRegression().fit(x_train, y_train)
```

```
model_knn = KNeighborsClassifier().fit(x_train, y_train)

cv_nb = cross_val_score(model_nb, x_test, y_test, cv=5)
cv_lr = cross_val_score(model_lr, x_test, y_test, cv=5)
cv_knn = cross_val_score(model_knn, x_test, y_test, cv=5)

print('Cross Validation=>')
print('Naive Bayes : ', cv_nb)
print('Logistic Regression : ', cv_lr)
print('KNN : ', cv_knn)
```

- ***Code*** **diatas merupakan tahap *cross validation* pada algoritma *naïve bayes, logistic regression,* dan KNN**

```
import matplotlib.pyplot as plt
%matplotlib inline
```

- **"`import matplotlib.pyplot as plt`" merupakan penginstalan *library matplotlib* berfungsi untuk menggambarkan hasil grafik pada tiap – tiap algoritma**

```
plt.plot(cv_nb)
plt.xlabel('Nilai K untuk Naive Bayes')
plt.ylabel('Akurasi Validasi Silang')


plt.plot(cv_lr)
plt.xlabel('Nilai K untuk Logistic Regression')
plt.ylabel('Akurasi Validasi Silang')


plt.plot(cv_knn)
plt.xlabel('Nilai K untuk KNN')
plt.ylabel('Akurasi Validasi Silang')
```

- ***Code*** **diatas merupakan pembuatan gambaran grafik pada tiap – tiap algoritma.**

```
metodeBN = MultinomialNB().fit(x_train, y_train)
metodeDTC = DecisionTreeClassifier().fit(x_train,y_train)
```

```
metodeKNN = KNeighborsClassifier().fit(x_train,y_train)
metodeLR = LogisticRegression().fit(x_train,y_train)

predictNB = metodeBN.predict(x_test)
predictDTC = metodeDTC.predict(x_test)
predictKNN = metodeDTC.predict(x_test)
predictLR = metodeLR.predict(x_test)

print('Accuracy=>')
print('Naive Bayes : ', metodeBN.score(x_test, y_test))
print('KNN : ', metodeKNN.score(x_test, y_test))
print('Logistic Regression : ', metodeLR.score(x_test, y_test))
```

- ***Code*** **diatas merpakan pembuatan hasil akurasi pada tiap – tiap algitma**

Lampiran 6: Evaluasi model algoritma KNN

```
myclassifier =
KNeighborsClassifier(n_neighbors=1,weights='uniform',p=2)
myclassifier.fit(x_train,y_train)


y_pred = myclassifier.predict(x_test)
y_pred


import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix,
ConfusionMatrixDisplay

cm = confusion_matrix(y_test, y_pred)
cm

disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=myclassifier.classes_)

disp.plot()
plt.show()
```

- ***Code* diatas merupakan tahap *confusion matrics* pada algoritma KNN**

```
myclassifier.classes_
```

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

- ***Code* diatas merupakan tahap pengukuran *Performance Vector* pada algoritma KNN**

Lampiran 7: Evaluasi model algoritma *naïve bayes*

```
NaiveBayes = MultinomialNB()
NaiveBayes.fit(x_train,y_train)
```

```
y_nb = NaiveBayes.predict(x_test)
y_nb
```

```
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix,
ConfusionMatrixDisplay
```

```
cm = confusion_matrix(y_test, y_nb)
cm
```

```
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=myclassifier.classes_)

disp.plot()
plt.show()
```

- ***Code* diatas merupakan tahap *confusion matrics* pada algoritma *naïve bayes***

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_nb))
```

- ***Code*** **diatas merupakan tahap pengukuran** ***Performance Vector*** **pada algoritma** ***naïve bayes***

Lampiran 8: Evaluasi model algoritma ***logistic regression***

```
LogisticRegression = LogisticRegression()
LogisticRegression.fit(x_train,y_train)


y_lr = LogisticRegression.predict(x_test)
y_lr


import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix,
ConfusionMatrixDisplay

cm = confusion_matrix(y_test, y_lr)
cm


disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=myclassifier.classes_)

disp.plot()
plt.show()
```

- ***Code*** **diatas merupakan tahap** ***confusion matrics*** **pada algoritma** ***logistic regression***

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_lr))
```

- ***Code*** **diatas merupakan tahap pengukuran** ***Performance Vector*** **pada algoritma** ***logistic regression***

Lampiran 9: *word cloud* (visualisasi hasil)

```
!pip install tweet-preprocessor
!pip install textblob
!pip install sastrawi
!pip install emoji
```

```
import os
import pandas as pd
import tweepy
import re
import string
from textblob import TextBlob
import preprocessor as p
from preprocessor.api import clean, tokenize, parse
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import datetime
from datetime import timedelta
import numpy as np
import emoji
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.feature_extraction.text import CountVectorizer
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
from Sastrawi.StopWordRemover.StopWordRemoverFactory import
StopWordRemoverFactory
```

```
import pandas as pd
def load_data():
  data = pd.read_csv('/content/hasil_LexiconBased.csv')
  return data
```

```
tweet_df = load_data()
tweet_df.head(1000)
```

```
s = pd.value_counts(tweet_df['Sentiments'])
ax = s.plot.bar()
n = len(tweet_df.index)
print (n)
for p in ax.patches:
  ax.annotate(str(round(p.get_height() / n * 100, 2)) + '%',
(p.get_x() * 1.005, p.get_height() * 1.005))
```

- ***Code*** **diatas merupakan pembuatan grafik untuk memaparkan jumlah sentimen analisis pada dataset yang sudah di analisis sentimen.**

```
wordcloud = WordCloud(width = 800, height = 800, background_color =
'black', max_words = 1000, min_font_size =
20).generate(str(tweet_df))

fig = plt.figure(figsize = (8,8), facecolor = None)
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
```

- ***Code*** **diatas merupakan pembuatan** ***word cloud*** **untuk mempresentasikan jumlah kata teks yang sering muncul dan hasil tersebut akan di buat ke dalam visual**