SKRIPSI

ANALISIS SENTIMEN TANGGAPAN MASYARAKAT KEPADA VIDEO YOUTUBE MENGENAI RESESI 2023 MENGGUNAKAN METODE NAÏVE BAYES DAN K-NEAREST NEIGHBOR (KNN)

Universitas Islam Negeri
SYARIF HIDAYATULLAH JAKARTA

Disusun Oleh :

ASSIDDIQIE ELZA PUTRA

11160930000098

PROGRAM STUDI SISTEM INFORMASI

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SYARIF HIDAYATULLAH

JAKARTA

2023 M / 1444 H

SKRIPSI

ANALISIS SENTIMEN TANGGAPAN MASYARAKAT KEPADA VIDEO YOUTUBE MENGENAI RESESI 2023 MENGGUNAKAN METODE NAÏVE BAYES DAN K-NEAREST NEIGHBOR (KNN)

ASSIDDIQIE ELZA PUTRA

11160930000098

PROGRAM STUDI SISTEM INFORMASI

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SYARIF HIDAYATULLAH

JAKARTA

2023 M / 1444 H

LEMBAR PERNYATAAN

DENGAN INI SAYA MENYATAKAN BAHWA SKRIPSI INI BENAR-BENAR HASIL KARYA SENDIRI YANG BELUM PERNAH DIAJUKAN SEBAGAI SKRIPSI ATAU KARYA ILMIAH PADA PERGURUAN TINGGI MANAPUN.

Jakarta, 5 April 2023

ASSIDDIQIE ELZA PUTRA
11160930000098

ABSTRAK

Assiddiqie Elza Putra – 11160930000098, Analisis Sentimen Tanggapan Masyarakat Kepada Video YouTube Mengenai Resesi 2023 Menggunakan Metode Naive Bayes Classifier (NBC) dan K-Nearest Neighbor (K-NN) di bawah bimbingan Zulfiandri dan A'ang Subiyakto.

YouTube merupakan salah satu media sosial yang diminati karena kemudahannya berbagi informasi melalui video sehingga para pengguna dapat dengan cepat mengetahui suatu *trend* atau topik terbaru yang terjadi. Akhir tahun 2022 salah satu topik yang dibicarakan adalah resesi di tahun 2023. Salah satu *content creator* YouTube yaitu Raymond Chin membahas topik ini melalui videonya yang berjudul "2023: Menuju KEHANCURAN DUNIA". Timbul beragam tanggapan positif hingga negatif. Penelitian ini merupakan penelitian kuantitatif dengan tipe *fine-grained sentiment analysis*, yaitu mengelompokkan respon atau pendapat ke dalam kategori positif atau negatif untuk mengekstrak sentimen dari komentar video YouTube tersebut. Penelitian ini menggunakan alur Metode SEMMA (Sample, Explore, Modify, Model, dan Assess). YouTube API dilakukan dalam *web scrapping* menggunakan Python. Peneliti menerapkan metode *lexicon based* untuk mendapatkan kelas sentimen pada dataset, yang selanjutnya dataset diklasifikasi menggunakan metode Naïve Bayes *Classifier* dan K-Nearest Neighbor, terakhir penelitian ini menggunakan *confusion matrix* dan *k-fold cross validation*. Metode Naïve Bayes Classifier menghasilkan *hyperparameter* alpha 100 dengan akurasi 69.43% sementara itu metode K-Nearest Neighbor didapatkan optimal k=11 dan menghasilkan akurasi 65.45%. Penelitian ini menunjukkan bahwa sentimen masyarakat terhadap video YouTube yang berjudul "2023: Menuju KEHANCURAN DUNIA", menghasilkan lebih banyak komentar bersentimen positif. Penelitian ini diharapkan dapat dimanfaatkan dalam penggalian opini untuk memahami pengguna video YouTube (*users*) kebanyakan khususnya penonton video YouTube Indonesia yang bertema konten sensitif, dalam hal ini terjadinya resesi disuatu negara.

Kata Kunci: *Fine-Grained Sentiment Analysis, Lexicon Based,* , K-Nearest Neighbor, Naïve Bayes Classifier, Confusion Matrix, YouTube, Resesi, Raymond Chin.

V BAB + xxxvii Halaman + 103 Halaman + 44 Gambar + 17 Tabel + Daftar Pustaka + Lampiran

Pustaka Acuan (56, 2015 – 2022)

KATA PENGANTAR

Assalamu'alaikum Warahmatullaah Wabarakaatuh

Alhamdulillah hirobbil alamin puji syukur peneliti panjatkan kepada Allah SWT, karena atas nikmat dan rahmat-Nya lah peneliti dapat menyelesaikan penelitian skripsi ini yang berjudul "**Analisis Sentimen Tanggapan Masyarakat Kepada Video Youtube Mengenai Resesi 2023 Menggunakan Metode *Naïve Bayes* dan K-*Nearest Neighbor* (KNN)**". Shalawat serta salam tidak lupa tercurahkan kepada junjungan nabi besar, Rasulullah Muhammad SAW beserta keluarga, sahabat, dan para pengikutnya hingga akhir zaman.

Skripsi ini dilakukan untuk memenuhi salah satu dari syarat untuk mendapatkan gelar Sarjana pada Program Studi Sistem Informasi Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta. Selama penyusunan skripsi ini tentunya peneliti mendapatkan banyak bantuan saran, motivasi, saran, dorongan hingga bimbingan dari berbagai pihak. Oleh karena itu, dengan segala hormat dan kerendahan hati, ucapan terima kasih secara khusus penulisa berikan kepada :

1. Bapak Husni Teja Sukmana, S.T., M.Sc, Ph.D. selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta.
2. Ibu Dr. Qurrotul Aini, M.T. selaku Ketua Program Studi Sistem Informasi Fakultas Sains dan Teknologi.

3. Bapak Zulfiandri, M.M.S.I. selaku dosen pembimbing 1 dan Bapak A'ang Subiyakto, Ph.D. selaku dosen pembimbing 2 yang telah membimbing peneliti dengan sabar dan telah meluangkan waktunya untuk membantu serta memotivasi kepada peneliti dalam penyelesaian skripsi ini.
4. Seluruh dosen Program Studi Sistem Informasi Universitas Islam Negeri Syarif Hidayatullah Jakarta yang telah memberikan waktu serta ilmu selama perkuliahan.
5. Orang tua serta adik – adik peneliti yang selalu memberikan dukungan dalam bentuk apapun, baik mental, fisik, finansial, maupun doa yang terus dilakukan hingga sampai detik ini.
6. Kepada sahabat SMA Gema Assidik, Dimas Tri M, Bintang, dan terutama Zulfikar Chamim dan Setyo Hadi J yang telah memberikan saran, masukan hingga doa kepada peneliti selama laporan skripsi ini.
7. Kepada sahabat pengerjaan skripsi Getar Nuansa R, Nurul Izza AR, Shaqila Erbeliza, Dhiyaan R, dan terutama Dhiya Alhaqqi yang telah memberikan motivasi, dorongan, semangat secara langsung maupun tidak langsung, sehingga peneliti dapat menyelesaikan laporan skripsi ini.
8. Wakhid Afifi yang telah memberikan banyak saran dan masukan hingga ide penyelesaian masalah kepada peneliti selama pembuatan laporan skripsi.

9. Frisky Moraza yang telah memberikan banyak sekali masukan hingga saran – saran kritis dalam membantu peneliti menyelesaikan laporan skirpsi ini.
10. Teman-teman Himatep yang secara tulus menghibur dan memberikan saran kepada peneliti dikala peneliti merasakan penat.
11. Seluruh pihak yang secara langsung ataupun tidak langsung dalam membantu peneliti menyelesaikan laporan skripsi ini dan memberikan doa yang tulus kepada peneliti. Meski tidak tertulis namun tidak mengurangi rasa hormat serta terima kasih dari peneliti.

Peneliti sangat menyadari bahwa dalam penyusunan laporan skripsi ini masih terdapat banyak kekurangan karena keterbatasan pengetahuan, wawasan, dan pengalaman yang peneliti miliki. Untuk itu peneliti tidak menutup diri terhadap segala bentuk saran maupun kritik yang bersifat membangun untuk kedepannya. Akhir kata peneliti berharap laporan skripsi ini dapat bermanfaat bagi pembaca dan peneliti sendiri.

Jakarta, Juni 2023

Assiddiqie Elza Putra

11160930000098

DAFTAR ISI

DAFTAR GAMBAR

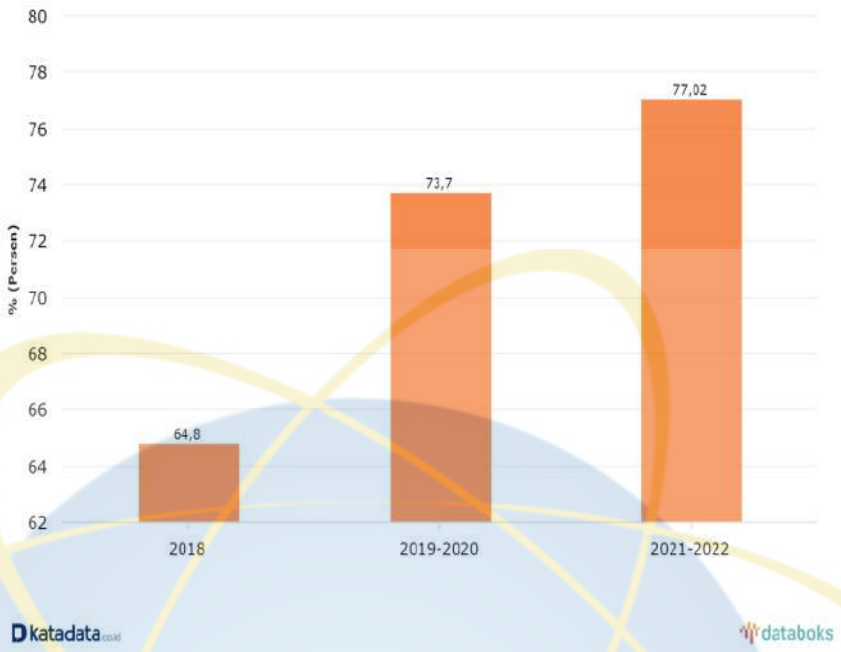
DAFTAR TABEL

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Berkembangnya peradaban manusia berbanding lurus dengan berkembangnya teknologi. Sekarang ini, teknologi sangatlah berkembang dimana banyaknya media yang dapat digunakan untuk berkomunikasi dengan tersambung ke jaringan internet. Selain berkomunikasi, seseorang juga dapat dengan mudah mencari hiburan melalui jaringan internet. Asosiasi Penyelenggara Jasa Internet Indonesia (APJII) beberapa waktu yang lalu merilis laporan profil internet Indonesia tahun 2022. Laporan ini menyatakan bahwa penetrasi penggunaan internet di Indonesia mengalami peningkatan dari tahun ke tahun. Dimana pada tahun 2018, penetrasi internet di Indonesia sampai pada 64.8% yang kemudian presentase ditahun berikutnya yaitu tahun 2019-2020 meningkat menjadi 73.7 % dan pada tahun 2021-2022 peningkatan terjadi hingga presentase penetrasi internet Indonesia di level 77.02%. Sumbangsih terbesar penggunaan internet terpusat di pulau Jawa dengan presentase 43.92%. Pada peringkat kedua Sumatra dengan presentase 16.63%. Kemudian, 5.53% penggunaan berasal dari Sulawesi, 4.88% berasal dari Kalimantan, 2.71% berasal dari Nusa Tenggara, 1.38% berasal dari Papua, 1.17% berasal dari Bali dan 0.81% berasal dari Maluku (Pahlevi 2022).



Gambar 1.1 Data Penetrasi Internet Indonesia

(Sumber : (Pahlevi 2022))

Peningkatan penetrasi internet di Indonesia didukung dengan adanya *smartphone*. Perangkat *smartphone* (telepon pintar) ini dapat dengan mudah digunakan oleh seseorang atau umumnya oleh masyarakat untuk mengakses media informasi maupun media komunikasi (Zulqornain, Indriati, and Putra 2021). Media – media tersebut diantaranya seperti Youtube, Instagram, Twitter, dll. YouTube merupakan salah satu media yang sangat popular, meningkatnya pupolaritas YouTube ini dikarenakan meningkatnya nilai guna berbagi video pada situs ini oleh para penggunanya (Mujianto 2019).

Youtube merupakan salah satu media sosial yang sangat diminati oleh banyak orang khususnya masyarakat Indonesia. Popularitas Youtube diprediksi akan terus meningkat dengan seiring bertambahnya pengguna. Pada pertengahan tahun 2017 Youtube mencatatkan jumlah penonton terdaftar sebanyak 1,5 miliar *user* (*logged-in*

monthly users). Pada tahun 2019 lembaga riset pasar statista sudah memprediksikan bahwa pengguna (*users*) Youtube ini bisa mencapai hingga 1,8 miliar ditahun 2021 (Mujianto 2019). Salah satu hasil riset menuturkan bahwa 92% pengguna Indonesia menyatakan Youtube adalah tujuan utama mereka dalam mencari media informasi dengan konten video (Arsiana 2021). Banyaknya pengguna aktif Youtube di Indonesia membuat peluang *content creator* Indonesia menjadi sangat besar dalam memanfaatkan Youtube sebagai *platform* untuk *marketing*. Selain untuk memasarkan produk, Youtube juga bisa menjadi *platform* untuk hiburan, media informasi, hingga memberikan ide ide tertentu. Beberapa *content creator* Youtube Indonesia diantaranya Kimbab *Family* yang berisikan tentang keluarga, Rusman dan *InYourDream* yang berisikan tentang game spesifik dota2, Kok Bisa! dan *insight* tentang ilmu informasi umum, Raymond Chin berisikan presepsi tentang suatu isu yang sedang terjadi hingga yang akan terjadi.

Raymond Chin, merupakan salah satu *content creator* Indonesia yang membahas tentang presepsi suatu isu yang sedang terjadi maupun yang akan terjadi khususnya mengenai bisnis dan ekonomi. Bergabung di Youtube sejak 23 Januari 2014, Raymond Chin mempunyai jumlah *subscriber* atau yang mengikuti kanal YouTubenya hingga lebih dari 1 juta pengguna. Sebagai *CEO & Founder* Ternak Uang, ia sangat memperhatikan isu isu ekonomi dan bisnis yang terjadi di Indonesia. Baik untuk marketing produknya maupun memberikan pengetahuan dan informasi pada masyarakat luas. Salah satu video atau *content* yang ia buat adalah “2023: Menuju KEHANCURAN DUNIA”. Video ini menjadi perbincangan yang hangat diawal bulan oktober 2022. Pasalnya banyak masyarakat yang menonton video tersebut hingga video tersebut diputar lebih dari 5,7 juta kali dengan jumlah *comment*

hingga 13.226 komentar. Video tersebut berisikan mengenai resesi tahun 2023 yang mengundang banyak sekali komentar positif netral hingga negatif.

Resesi menjadi topik yang sangat dibicarakan diakhir tahun 2022. Seperti yang sudah dijelaskan sebelumnya, bahwa salah satu *content creator* yang membahas isu resesi ini adalah Raymond Chin. Kondisi saat terjadinya penurunan produk domestik bruto selama 2 kuartal secara berulang dalam satu tahun berjalan merupakan kondisi yang dapat disebut resesi (Blandina, Noor Fitrian, and Septiyani 2020). Melemahnya perekonomian global yang kemudian mempengaruhi ekonomi domestic negara lain diseluruh dunia merupakan tanda-tanda dari resesi. Terdapat beberapa hal yang disebabkan karena adanya resesi ekonomi, diantaranya penurunan aktivitas- aktivitas ekonomi seperti lapangan kerja, investasi dan keuntungan perusahaan dalam waktu bersamaan. Biasanya deflasi (penurunan harga), atau bahkan inflasi (kenaikan harga) berkaitan dengan resesi ekonomi yang dapat disebut stagflasi (Blandina et al. 2020). Salah satu faktor terjadinya resesi adalah tingkat pengangguran yang semakin tinggi.

Kombinasi antara resesi yang akan terjadi dengan video yang dibuat oleh Raymond Chin yang membahas tentang adanya resesi pada tahun 2023 merupakan hal yang sangat menarik perhatian *public* terutama masyarakat Indonesia. Banyaknya komentar yang diberikan oleh *public* mengandung banyak tanggapan, dan tanggapan tersebut dapat dianalisa menggunakan suatu teknik untuk mengetahui sentimen atau tanggapan kebanyakan masyarakat di Indonesia yaitu menggunakan analisa sentimen. Analisa sentimen atau yang bisa disebut (*opinion mining*), merupakan suatu proses mengolah, mengekstrak, dan memahami data tekstual secara otomatis untuk mendapatkan suatu kandungan informasi sentimen dalam suatu kalimat atau opini (Buntoro 2017).

Penelitian mengenai analisa sentimen telah dilakukan oleh beberapa peneliti seperti oleh Samsir et al pada tahun 2021. Penelitian tersebut berjudul "Analisis Sentimen Pembelajaran Daring Pada Twitter dimasa Pandemi COVID-19 Menggunakan Metode Naïve Bayes". Penelitian tersebut mengenai respon dari masyarakat mengenai perubahan mendadak penerapan pembelajaran daring mengingat adanya pandemi Covid-19. Pembelajaran daring yang awalnya merupakan strategi justru menjadi kontroversi karena singkatnya proses adaptasi. Tujuan dari penelitiannya adalah menganalisis opini yang ada dipublik terhadap pembelajaran daring dimasa *pandemic* Covid-19 di Indonesia diakhir tahun 2020 yang berasal dari twitter. Penelitian tersebut menggunakan algoritma Naïve Bayes dalam analisisnya dengan hasil bahwa pembelajaran daring hanya memiliki 30% sentimen positif, 1% netral dan 69% sentimen negative pada periode November 2020. Dengan hasil sentimen negatif yang sangat tinggi maka masyarakat tidak puas terhadap pembelajaran daring (Samsir et al. 2021) .

Kemudian terdapat penelitian analisa sentimen dari Deviyanto dan Wahyudi pada tahun 2018 yang berjudul "Penerapan Analisis Sentimen Pada Pengguna Twitter Menggunakan Metode K-Nearest Neighbor". Penelitian tersebut menjelaskan mengenai analisis sentimen masyarakat Indonesia terhadap topik pilkada DKI tahun 2017. Data yang digunakan terdapat 2000 data tweet berbahasa Indonesia yang terkumpul pada bulan Januari tahun 2017. Pengklasifikasian nilai sentimen terbagi menjadi positif dan negatif dengan menggunakan algoritma KNN dengan pembobotan kata TF-IDF dan fungsi *Consine Similarity*. Penelitian tersebut menghasilkan nilai akurasi terbesar adalah 67,5% ketika k=5, presisi tertinggi 56,94% ketika k=5, dan *recall* 78,24% dengan k=15 (Deviyanto and Wahyudi 2018).

Selanjutnya terdapat penelitian dari Zulqornain, Indriati, dan Putra pada tahun 2021 dengan judul penelitian "Analisis Sentimen Tanggapan Masyarakat Aplikasi Tiktok Menggunakan Metode Naïve Bayes dan Categorial Propotional Difference (CPD)". Penelitian tersebut meneliti mengenai aplikasi tiktok yang dapat diakses oleh semua orang bahkan dari segala umur. Maka dari itu peneliti Zulqornain et al., melakukan analisis sentimen terhadap ulasan aplikasi TikTok untuk membantu para orang tua dalam memilih aplikasi yang digunakan oleh anaknya. Penelitian tersebut menggunakan metode Naïve Bayes dan *Categorial Propotional Difference*. Variasi *term* digunakan dengan menggunakan 5-*cross Validation*. Penelitian tersebut mendapatkan hasil yang maksimal dengan menggunakan 100% *term* untuk pengujian dengan nilai akurasi 0,729947, kemudian nilai presisi 0,745854, nilai *recall* 0,926118, dan nilai f-*measure* 0,824511 (Zulqornain et al. 2021).

Dengan penelitian penelitian sebelumnya yang menggunakan beberapa metode yang ada seperti metode *Naïve Bayes*, K-*Nearest Neighbor*, *Categorial Propotional Difference*, *Support Vector Machine*, *Lexicon Based Features*, dsb. Peneliti dalam penelitian ini tertarik untuk menggunakan metode Naïve Bayes dan K-*Nearest Neighbor*. Dari referensi-referensi penelitian tersebut, penelitian ini akan membandingkan metode Naïve Bayes dan K-*Nearest Neighbor* dalam menganalisis sentimen masyarakat.

Penelitian ini menggunakan metode Naïve Bayes dikarenakan metode ini dapat menghasilkan akurasi yang maksimal walaupun dengan data latih yang sedikit. *Naïve Bayes Classifier* (NBC) sendiri merupakan salah satu dari beberapa metode alasan probabilitas (*probabilistic reasoning*) dimana metode ini dari dataset yang ada,

dihitung sekumpulan probabilitasnya dengan mengkombinasikan nilai dan menjumlahkan frekuensi yang ada (Tempola, Muhammad, and Khairan 2018).

Kemudian untuk metode lainnya, penelitian ini menggunakan K-*Nearest Neighbor*, metode ini digunakan karena metode ini tangguh terhadap data *noise*. Menurut salah satu penelitian yaitu penelitian Puspita dan Widodo tahun 2020, Algoritma KNN ini merupakan algoritma yang sudah *popular* (Puspita and Widodo 2021). Lalu algoritma ini pun mempunyai beberapa keunggulan seperti algoritma yang cepat, mudah dimengerti, pelatihan yang sederhana dan efektif jika dataset ukuran yang cukup besar.

Dengan alasan alasan diatas, penelitian ini akan dilakukan untuk menganalisis opini-opini masyarakat terhadap video Raymond Chin mengenai resesi pada tahun 2023 pada *platform* Youtube dengan menggunakan metode Naïve Bayes dan K-*Nearest Neighbor* sebagai topik penelitian skripsi yang berjudul "**Analisis Sentimen Tanggapan Masyarakat Mengenai Resesi 2023 Menggunakan Metode Naïve Bayes dan K-*Nearest Neighbor*".

1.2 Identifikasi Masalah

Berdasarkan latar belakang permasalahan diatas, penulis mengidentifikasikan permasalahan yang akan dijelaskan berikut :

1. Membandingkan hasil dari 2 metode yang digunakan yaitu metode *Naïve Bayes* dan K-*Nearest Neighbor*.
2. Banyaknya komentar yang diberikan oleh masyarakat terhadap video Raymond Chin mengenai resesi 2023 ini menimbulkan banyak pro dan kontra. Sekaligus mengemukakan opini-opini yang diutarakan oleh pengguna

Youtube Indonesia terhadap video Youtube “2023: Menuju KEHANCURAN DUNIA” mengenai resesi 2023.

3. Belum diketahui sentimen pengguna Youtube Indonesia mengenai video youtube mengenai resesi pada tahun 2023 dari Raymond Chin yang berjudul “2023: Menuju KEHANCURAN DUNIA” berdasarkan komentar yang ada.

1.3 Rumusan Masalah

Berdasarkan identifikasi masalah diatas, dapat dirumuskan masalah dalam penelitian ini yaitu bagaimana analisis sentimen komentar dari pengguna Youtube Indonesia terhadap video Youtube menggunakan metode Naïve Bayes dan K-*Nearest Neighbor*?

1.4 Batasan Masalah

Berdasarkan masalah yang telah dirumuskan diatas maka ruang lingkup masalah dibatasi pada :

1. Analisis sentimen dilakukan menggunakan 2 metode yaitu Naïve Bayes dan K-*Nearest Neighbor*.
2. Data yang digunakan dalam penelitian ini adalah komentar berbahasa Indonesia video youtube berjudul “2023: Menuju KEHANCURAN DUNIA” milik Raymond Chin yang bertemakan resesi 2023.
3. Data akan diklasifikasikan menjadi 2 bagian yaitu positif dan negative
4. Perangkat keras yang digunakan adalah laptop HP-D14MGQ2 dengan Ram 8 GB yang terhubung ke internet.
5. Perangkat lunak yang digunakan OS Windows 10 Pro, Rapid Miner versi 10.0.0, Python versi 3.10.9, Jupyter *Notebook* dan Microsoft Excel.

1.5 Tujuan Penelitian

Tujuan umum pada penelitian ini adalah untuk menganalisis dan mengemplementasikan sentimen topik menggunakan metode KNN dan Naïve Bayes pada media sosial Youtube video yang berjudul "2023: Menuju KEHANCURAN DUNIA" mengenai resesi 2023 terhadap tanggapan masyarakat yang nantinya akan menjadi acuan dan rekomendasi kepada konten video yang serupa. Kemudian tujuan khusus dari penelitian ini adalah :

a. Membandingkan performa dari metode Naïve Bayes dan K-Nearest Neighbor.
b. Mengetahui kecenderungan opini dari pengguna YouTube Indonesia pada komentar video YouTube Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA" mengenai resesi 2023.
c. Memberikan saran lanjutan untuk video dan topik serupa berdasarkan hasil sentimen dari pengguna YouTube Indonesia untuk video dan topik serupa.
d. Menampilkan opini - opini yang diutarakan oleh kebanyakan pengguna YouTube Indonesia terhadap video YouTube Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA" mengenai resesi 2023.

1.6 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat kepada beberapa pihak, sebagai berikut

1.6.1 Bagi peneliti:

1. Dapat memberikan pemahaman lebih mengenai proses analisa sentimen menggunakan metode Naïve Bayes dan K-*Nearest Neighbor*.
2. Sebagai bahan referensi untuk penelitian selanjutnya mengenai analisis sentimen.

1.6.2 Bagi universitas:

1. Sebagai bahan referensi untuk penelitian selanjutnya mengenai analisis sentimen yang menggunakan metode Naïve Bayes dan K-*Nearest Neighbor*.
2. Sebagai bahan evaluasi dalam mengembangkan keilmuan dalam menganalisis sentimen masyarakat Indonesia.

1.6.3 Bagi Umum:

1. Memberikan wawasan kepada peneliti maupun pembaca dalam topik analisis sentimen menggunakan metode yang digunakan.
2. Memberikan informasi kepada pembaca bahwa video mengenai resesi 2023 yang dibahas oleh Raymond Chin ditangkap baik atau tidak oleh pangguna youtube Indonesia dan masyarakat Indonesia pada umumnya.

1.6.4 Bagi *Content Creator*:

1. Memberikan pengetahuan kepada content creator bahwa kebanyakan penonton pada video ini memiliki lebih banyak sentimen positif dan negative.
2. Menjadi acuan kepada masyarakat yang ingin membuat video serupa (*content creator*), jika video bernilai positif maka dapat menjadi inspirasi untuk membuat tema serupa dengan penyampaian yang baik, namun jika sebaliknya maka video ini tidak baik untuk menjadi acuan dan rekomendasi.

1.7 Metodologi Penelitian

Penelitian ini mempunyai beberapa tahapan dalam pembuatannya yaitu :

1.7.1 Studi Literatur

Pada studi literatur peneliti melakukan *research* dan mempelajari materi materi terkait penelitian analisis sentimen dari referensi yang terpercaya. Informasi-informasi yang didapatkan tersebut akan digunakan dalam penyusunan penelitian ini. Semua studi literatur yang dijadikan acuan atau referensi dapat dilihat pada halaman daftar Pustaka.

1.7.2 Pengumpulan Data

Pada tahap ini dilakukannya pengumpulan data komentar video Youtube Raymond Chin yang berjudul “2023: Menuju KEHANCURAN DUNIA”. Pengumpulan data menggunakan bahasa

pemrograman pyhton yang akan dimasukkan kedalam Microsoft Excel.

1.7.3 Metode SEMMA

Metode yang dilakukan pada penelitian ini menggunakan SEMMA *Data Mining Process* (Alizah, 2020). Terdapat beberapa tahapan yang akan dilakukan, Adapun tahapan tersebut sebagai berikut :

i. *Sample*
ii. *Explore*
iii. *Modify*
iv. *Model*
v. *Assess*

1.7.4 Proses *Cross Validation*

Tahap ini peneliti mencari akurasi dari masing-masih metode yaitu Naïve Bayes dan K-*Nearest Neighbor* dengan membagi 2 data menjadi data *training* dan *testing*.

1.8 Sistematika Penulisan

Dalam penyusunan laporan penelitian, pembahasan terbagi dalam lima bab yang secara singkat akan diuraikan sebagai berikut :

BAB 1 PENDAHULUAN

Bab ini berisi penjelasan secara singkat mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan

dan manfaat penelitian, metode penelitian, dan sistematika penulisan

BAB 2 LANDASAN TEORI

Bab ini membahas mengenai dasar-dasar teori yang mendukung penelitian ini, dimulai dari sejarah hingga hal-hal yang berhubungan dengan masalah yang ditulis dan pedoman untuk menyelesaikan masalah yang ada.

BAB 3 METODOLOGI PENELITIAN

Bab ini membahas tentang metodologi yang digunakan dalam penelitian ini.

BAB 4 PEMBAHASAN DAN HASIL

Bab ini membahas mengenai hasil yang didapat dari penelitian analisis sentimen dengan metode Naïve Bayes dan K-*Nearest Neighbor*.

BAB 5 PENUTUP

Bab ini berisi kesimpulan yang berkenaan dengan hasil pemecahan masalah beserta saran untuk penyempurnaan dan pengembangan selanjutnya.

DAFTAR PUSTAKA

BAB 2

LANDASAN TEORI

2.1 Analisis Sentimen

Menurut penelitian yang dilakukan oleh Buntoro pada tahun 2017, proses mengekstrak, memahami dan mengolah data tekstual dengan cara otomatis dalam mendapatkan suatu informasi sentimen yang terkandung pada suatu kalimat opini merupakan analisis sentimen atau yang dapat disebut *opinion mining* (Buntoro 2017). Kemudian menurut samsir et al, pada tahun 2021 didalam analisis sentimen, dilakukan *data mining* untuk memproses, menganalisis, dan mengekstrak data tekstual dalam suatu entitas seperti sebuah produk, layanan, individu, fenomena, atau topik lainnya (Samsir et al. 2021).

Dapat dikatakan Analisis Sentimen merupakan sebuah proses pengelompokan dan menentukan sentimen suatu polaritas teks dalam kalimat sehingga dapat ditentukannya kategori sebagai sentimen positif, negatif ataupun netral (Pratama et al. 2019; Ramadhan and Setiawan 2019). Saat ini, banyaknya penelitian dan aplikasi berbasis analisis sentimen dipengaruhi oleh besarnya manfaat serta pengaruh dari analisis sentimen yang kemudian topik ini berkembang pesat (Samsir et al. 2021).

Analisis sentimen dapat disebut dengan opinion mining karena dapat difokuskan kepada pendapat yang bernilai positif atau negatif. Jejaring sosial seperti twitter, Instagram, bahkan komentasr youtube dapat digunakan dalam analisis sentimen dimana sosial media tersebut dapat menentukan persepsi publik. Sentimen analisis merupakan percabangan dari data mining. Kemudian yang dimaksud dengan data mining itu sendiri adalah sebuah proses dimana informasi diekstrak yang pada

akhirnya menghasilkan suatu informasi yang berharga (Nurdin and Astika 2015). Dapat dikatakan jika data mining merupakan sebuah proses dalam pencarian informasi dengan menggunakan teknik tertentu. Terdapat banyak teknik dan metode yang ada pada data mining. Maka dari itu, pemilihan suatu teknik atau algoritma yang tepat akan sangat bergantung kapada tujuan penelitian yang diinginkan (Puspita and Widodo 2021).

Terdapat beberapa kelebihan dari sentimen analisis itu sendiri seperti misalnya sentimen analisis dapat memberikan hasil opini atau wawasan yang objektif. Dengan basis Artificial Intelligence (AI) alat analisis sentimen ini dapat menghindari bias pribadi dalam peninjauan manusia. Pada akhirnya akan mendapatkan hasil yang konsisten dan objektif dalam menganalisis suatu pendapat. Contoh kalimat dalam suatu komentar misalnya "Saya menyukai ide dan konsep dari video ini, namun kecewa pada penyampaiannya". Mungkin pemilik video akan mengabaikan bagian mengecewakan pada komentar atau ulasan video tersebut dan melakukan bias secara positif kepada ide dan konsep video tersebut. Namun, analisis sentiment mengklasifikasikan dan mengurutkan teks tersebut untuk menangkap emosi secara objektif. Selanjutnya analisis sentiment juga mempunyai beberapa tantangan. Pertama adalah suatu kalimat yang dapat menyatakan positif namun disisi lain dapat dianggap negatif. Tantangan kedua ialah seseorang tidak selalu menyampaikan pendapat dengan cara yang sama (Nagamma et al. 2015).

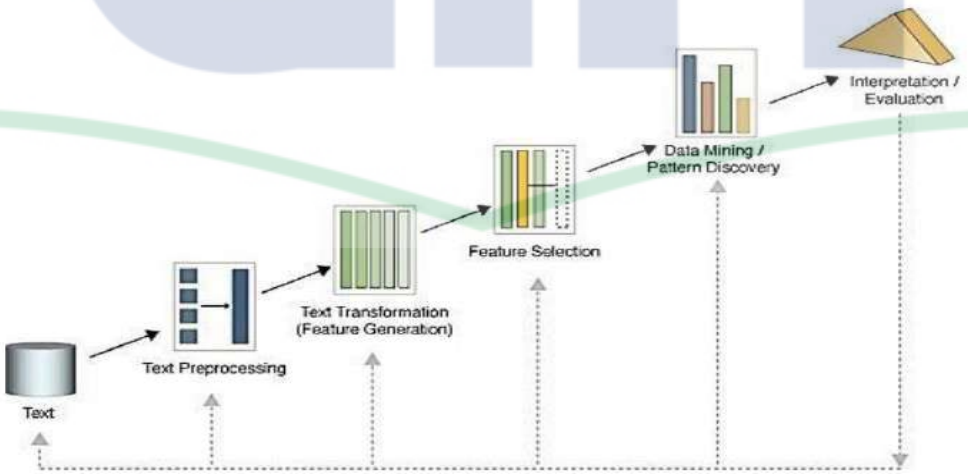
Analisis sentimen dapat membantu menemukan kata yang mengandung sentimen dan membantu dalam memahami hubungan ulasan tekstual tersebut dengan kosekuensi dari ulasan tersebut (Nagamma et al. 2015). Salah satu contohnya adalah

ulasan penjualan album musisi mempengaruhi posisi musisi tersebut dalam peringkat lagu/album *billboard music of the week*.

2.2 *Text Mining*

Analisis sentimen dan penambangan teks (*text mining*) adalah dua hal yang tidak dapat dipisahkan, terutama jika Analisis sentimen dilakukan pada media sosial. penambangan teks (*text mining*) adalah studi yang lebih rinci tentang penambangan data mengungkapkan pola tersembunyi dalam teks, yang kemudian analisis sentimen dan penambangan teks dikombinasikan dengan mengumpulkan pendapat, memproduksi alat bantuan yang sangat baik dan dapat diandalkan (Siringoringo and Jamaludin 2019).

Proses pencarian informasi dalam *text mining* dapat menghasilkan analisis emosi atau perasaan yang secara emosional mengidentifikasi pernyataan, apakah itu positif atau negatif. Objek *text* mining adalah dokumen yang tidak terstruktur atau semi-terstruktur. Penambangan teks atau *text mining* secara efisien mengekstrak suatu informasi yang diperlukan dari banyak dokumen (Samsir et al. 2021).



Gambar 2.1 Alur *Text Mining*

(Sumber : (Siringoringo and Jamaludin 2019))

Berdasarkan gambar diatas, alur *text mining* secara umum merupakan tahap pengumpulan teks, preprocessing atau pemrosesan awal teks, transformasi teks atau *text transformation*, seleksi fitur atau *feature selection*, *data mining* dan interpretasi.

2.3 Data *Pre-processing*

Perancangan penelitian ini dimulai dari melakukan pemrosesan awal atau *pre-processing*. Proses *pre-processing* dilakukan sebelum *dataset* dimasukkan kedalam model. Tahapan ini merupakan suatu tahap pemrosesan data dimana data diolah menjadi data yang siap untuk dianalisis. Setelah data tersebut sudah terstruktur maka dapat diolah dan diproses lebih lanjut. Untuk mendapatkan data yang baik, maka dilakukanlah beberapa teknik *preprocessing* yang digunakan peneliti (Puspita and Widodo 2021):

a. *Data Validation*

   Tahapan ini merupakan tahapan dimana peneliti mengidentifikasi sekaligus menghapus data yang tidak digunakan maupun data yang tidak konsisten terutama kepada data yang *missing.*

b. *Data Integration and Transformation*

   Kemudian pada tahapan ini, peneliti meningkatkan akurasi dari data yang didapatkan dari metode yang digunakan oleh peneliti.

c. *Data Size Redution and Dicretization*

   Selanjutnya pada tahapan ini data yang sebelumnya sudah terkumpul akan dirapihkan oleh peneliti. Proses ini dapat dikatakan juga proses *cleaning* dimana proses ini untuk mengurangi *noise* untuk menghapus kata kata yang tidak bermakna.

2.4 Klasifikasi

Klasifikasi merupakan salah satu tahap dan teknik dari *data mining*. Algoritma klasifikasi ini dipadukan dengan menggunakan RapidMiner pada penelitian ini. Klasifikasi adalah suatu proses yang dilakukan untuk menguji akurasi metode-metode yang digunakan pada penelitian tertentu dalam menentukan sentimen disuatu kalimat (Buntoro 2017).

Klasifikasi adalah salah satu tugas penambangan data (*data mining)* yang paling penting. Pengklasifikasian dibuat dari kumpulan data latih dengan kelas tertentu Klasifikasi adalah pengelompokan properti ke dalam kategori yang sesuai. Vektor fitur pelatihan tersedia dan kelas-kelasnya diketahui, kemudian vektor fitur pelatihan digunakan untuk mendesain klasifikasi (Wibawa et al. 2018). Proses klasifikasi dibagi menjadi kelas-kelas yang berbeda yang disebut pengklasifikasi berbasis keputusan (Gupta, Gupta, and Singh 2015).

Seperti yang sudah dijelaskan sebelumnya, terdapat banyak metode atau algoritma yang biasanya dipakai disuatu penelitian. Algoritma tersebut diantaranya Naïve Bayes, *Support Vector Machine, Decission Tree Support, Neural Network, Fuzzy, K-Nearest Neighbor,* dan sebagainnya. Tentunya terdapat kelebihan dan kelemahan yang dipunyai masing-masing metode.

Data mining metode klasifikasi melakukan prosesnya dengan belajar dengan data yang sudah ada, kemudian melakukan klasifikasi untuk data baru, hasil dari metode klasifikasi adalah categorical (nominal atau ordinal). Dalam melihat apakan estimasi akurasi yang diberikan oleh model klasifikasi benar, maka ada yang

dinamakan confusion matrix. Dari matrix tersebut para miner dapat melakukan estimasi akurasi dari proses yang sudah dijalankan.

Metode klasifikasi data mining melakukan proses dengan memeriksa data yang sudah ada dan kemudian mengklasifikasikan data baru. Hasil dari metode klasifikasi ini adalah sebuah kategori (nominal atau ordinal). Untuk menentukan apakah perkiraan akurasi yang diberikan oleh model klasifikasi benar, ada yang disebut dengan *confusion matrix*. Dari matriks ini, peneliti dalam penelitian *mining* dapat memperkirakan keakuratan proses yang sudah dijalankan.

		Observed	
		True	*False*
Predicted Class	*True*	*True Positive* (TP)	*False Positive* (FP)
	False	*False Negative* (FN)	*True Negative* (TN)

Gambar 2.2 *Confussion Matrix*

(Sumber : (Pratiwi, Handayani, and Sarjana 2021))

Pada tabel diatas dijelaskan bahwa TP adalah *true positive,* merupakan data positif yang terklasifikasi dengan benar oleh sistem. TN adalah *true negative,* merupakan data negatif yang terklasifikasi dengan benar oleh sistem. FN adalah *false negative,* merupakan data negatif yang terklasifikasi salah oleh sistem. TN adalah *true negative,* merupakan data positif yang terklasifikasi dengan benar oleh sistem. Dapat dikatakan bahwa nilai akurasi didapat dari perbandingan antara keseluruhan data dengan data yang terklasifikasi benar (Pratiwi et al. 2021). Dengan menggunakan

Confussion Matrix, nilai akurasi, presisi, *recall*, *error* dapat dihitung. Persamaan untuk nilai akurasi diperoleh dengan cara:

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN} \text{ X } 100\% \qquad (2.1)$$

Kemudian terdapat nilai presisi. Nilai presisi ini menggambarkan sejumlah data yang berkategorikan positif yang diklasifikasi secara benar dibagi oleh total yang diklasifikasi positif. Persamaan untuk nilai presisi diperoleh dengan cara:

$$\text{Presisi} = \frac{TP}{TP + FP} \text{ X } 100\% \qquad (2.2)$$

Selanjutnya terdapat nilai *recall*. Nilai *recall* memperlihatkan data yang berkategorikan positif yang terklasifikasi secara benar oleh sistem. Persamaan untuk nilai presisi ini diperoleh dengan cara :

$$\textit{Recall} = \frac{TP}{TP + FN} \text{ X } 100\% \qquad (2.3)$$

Nilai terakhir merupakan nilai *error*. Nilai *error* merupakan sejumlah data yang diidentifikasi salah, sehingga dapat terlihat besarnya tingkat kesalahan dari sistem. Persamaan untuk presentase nilai *error* dapat diperoleh dengan cara :

$$\textit{Error} = \frac{FP}{TP} \text{ X } 100\% \qquad (2.4)$$

Dengan persamaan persamaan inilah bisa diketahui nilai – nilai atau hasil dari nilai akurasi, presisi, *recall*, dan *error*.

2.4.1 *Naïve* Bayes

Naïve Bayes adalah salah satu metode klasifikasi berdasarkan pada teorema Bayes. Terdapat ciri penting pada metode Naïve Bayes, yaitu asumsi independensi yang sangat kuat (naif) dari masing – masing kejadian atau

kondisi (Wibawa et al. 2018). Naïve Bayes *Classifier* merupakan algoritma yang digunakan untuk mencari nilai probabilitas tertinggi untuk mengklasifikasikan data *test* pada kategori yang tepat. Teorema Bayes mempunyai rumus umum (Pratama et al. 2019):

$$P(H \mid Y) = \frac{P(X \mid H) \times P(H)}{P(X)} \quad (2.5)$$

Dimana :

i. P(H|X) adalah kemungkinan probabilitas akhir (probabilitas posterior) dari hipotesis H terjadi ketika bukti yang diberikan E terjadi.
ii. P(X|H) adalah probabilitas munculnya bukti E akan mempengaruhi hipotesis H.
iii. P(H) adalah hipotesis probabilitas sebelumnya H terjadi terlepas dari bukti apa pun.
iv. P(X) adalah bukti probabilitas sebelumnya dari E terlepas dari hipotesis atau bukti lainnya.

Kemudian dari persamaan diatas, persamaan Naïve Bayes tersebut dikembangkan lagi menjadi (Samsir et al. 2021):

$$P(H \mid X1,X2, \ldots Xn) = \frac{P(X1, X2, \ldots Xn \mid H) \times P(H)}{P(X1, X2, \ldots Xn)} = \frac{P(X1 \mid H) \times P(X2 \mid H) \ldots P(Xn \mid H) \times P(H)}{P(X1, X2, \ldots Xn)} \quad (2.6)$$

Dimana hasil dari persamaan diatas yaitu P (H | X1,X2, …. Xn) adalah hasil hitung dari semua probabilitas posterior pada nilai X untuk semua nilai H, yang kemudian algoritma Naïve Bayes ini membuat suatu prediksi yang didasari probabilitas maksimum dari probabiliras posterior laplace yang akan

ditampilkan pada persamaan dibawah ini, dimana c adalah nilai pada H (Dong et al. 2020):

$$P(Xi \mid H) = \frac{Nic + 1}{Nc + c} \qquad (2.7)$$

Keuntungan menggunakan Naive Bayes adalah hanya membutuhkan sedikit data pelatihan untuk menentukan rata-rata parameter dan varian dari variabel yang diperlukan untuk klasifikasi. Naive Bayes merupakan salah satu metode klasifikasi *supervised document* yang artinya membutuhkan data *training* sebelum melakukan proses klasifikasi. Pada proses pelatihan telah ditentukan kategori dokumen (data pelatihan) yang selanjutnya akan diolah dan membentuk pengetahuan berupa nilai probabilitas untuk setiap kata. Proses ini akan menghasilkan sebuah kata pada setiap dokumen yang mencirikan dokumen tersebut dalam kategori tertentu (Devita, Herwanto, and Wibawa 2018).

2.4.2 K-Nearest Neighbor

Metode K-*Nearest Neighbor* atau yang biasa disingkat dengan KNN merupakan teknik *lazy* learning. Yang dimaksud dengan *lazy learning* adalah metode ini digunakan dalam klasifikasi data yang jaraknya berdekatan (Puspita and Widodo 2021). Dengan begitu metode ini mempunyai prinsip kerja dengan mencari data dengan jarak terdekat yang selanjutnya akan dievaluasi dengan k tetangga (*neighbor*) didalam data pelatihan (Devita et al. 2018). Kemudian menurut penelitian lain metode KNN ini merupakan algoritma pembelajaran yang banyak digunakan pada sistem CPSS (*Cyber-*

physical-social systems) yang diperuntukan dalam menganalisis dan menambang data (Zhang et al. 2020).

K-Nearest *Neighbor* atau KNN merupakan *supervised method* yang dapat diartikan sebagai metode yang membutuhkan data latih untuk mengklasifikasikan objek yang terdekat (Devita et al. 2018). Perbedaan antara *supervised learning* dan *unsupervised learning* adalah bahwa *supervised learning* bertujuan untuk menemukan pola baru dalam data dengan menghubungkan pola data yang ada dengan data baru. Sedangkan pada *unsupervised learning*, data belum memiliki pola, dan tujuan *unsupervised learning* adalah untuk menemukan pola pada data tersebut (Liantoni 2016).

Algoritma KNN menggunakan klasifikasi tetangga sebagai nilai prediksi dari sampel data uji baru. Jarak yang digunakan adalah *Euclidean Distance*. Jarak *Euclidean* adalah jarak yang paling umum digunakan untuk data numerik. Algoritma KNN merupakan algoritma yang menentukan nilai jarak pada data testing dengan data *training* berdasarkan nilai terkecil dari nilai tetangga terdekat (Liantoni 2016). Dapat dikatakan metode KNN cukup sederhana, berdasarkan dengan jarak terpendek menuju ke *training sample* dari *query instance* untuk menentukan KNN.

Nilai k terbaik untuk algoritma ini bergantung pada data. Secara umum, nilai k yang tinggi mengurangi efek noise pada klasifikasi, tetapi membuat batas antara setiap klasifikasi menjadi lebih kabur. Nilai k yang baik dapat dipilih dengan optimasi parameter, misalnya dengan menggunakan cross-validation. Metode ini sendiri memiliki beberapa kelebihan seperti

misalnya metode ini tangguh terhadap data latih yang memiliki banyak *noise* dan cukup efektif apabila data latih memiliki data yang besar (Kustiyahningsih and Syafa'ah 2016).

2.5 Pembobotan Kata

Pembobotan kata memiliki tujuan untuk memberikan bobot kepada suatu kata berdasarkan frekuensi kemunculan kata. Dari fitur kata tersebut yang sudah diberi bobot, maka selanjutnya dapat digunakan untuk proses klasifikasi (Rofqoh, Perdana, and Fauzi 2017). Melakukan perhitungan bobot kata (*term*) dapat menggunakan metode TF-IDF. Metode TF-IDF adalah salah satu cara untuk menemukan bobot suatu kata (*term*) pada sebuah dokumen. *Term Frequency* (TF) adalah suatu pernyataan dimana tingkat keseringan (*frequency*) munculnya suatu kata (*term*) dalam suatu dokumen. Jika *Invers Document Frequency* (IDF) merupakan banyaknya jumlah dokumen disaat kata (*term*) tersebut muncul (Deviyanto and Wahyudi 2018). Atau dapat dikatakan perhitungan IDF digunakan untuk menghitung kuantitas *term* yang berfungsi sebagai ukuran tingkat signifikansi suatu *term* dalam suatu dokumen. Berikut perhitungan TF dan IDF dalam persamaan berikut (Devita et al. 2018):

$$TF\ (d, t) = f\ (d, t) \qquad (2.8)$$

$$IDF\ (t) = 1 + \log\left(\frac{Nd}{df\ (t)}\right) \qquad (2.9)$$

Dimana :

1. f (d, t) : Frekuensi kemunculan kata (*term*) t pada dokumen d
2. *Nd* : Jumlah seluruh dokumen
3. *df (t)* : Jumlah dokumen yang terdapat *term* t

Dengan persamaan tersebut dapat digunakan persamaan berikut untuk menemukan nilai TF-IDF

$$TF - IDF = TF(d, t).\ IDF(t) \quad (2.10)$$

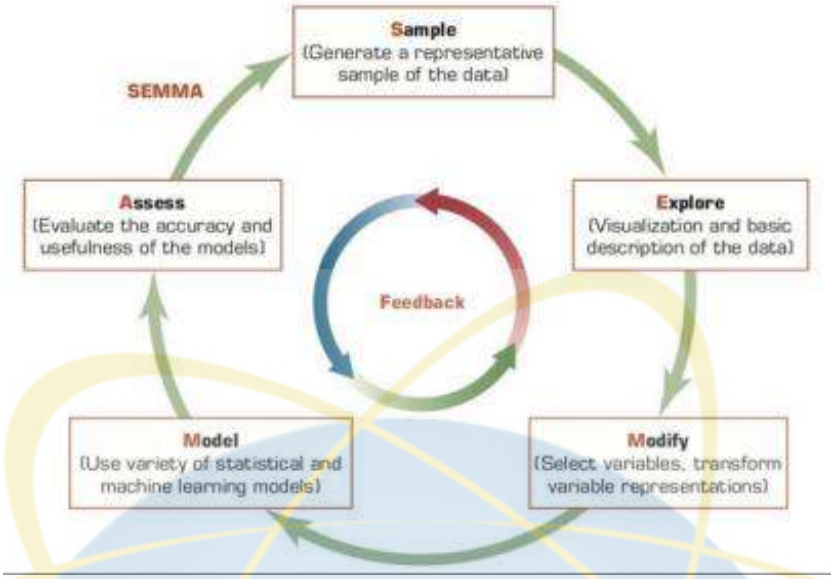
Selanjutnya menurut penelitian lain, metode Term Frequency Inverse Document Frequency (TF-IDF) adalah metode yang digunakan untuk menentukan seberapa jauh suatu kata (*term*) berhubungan dengan suatu dokumen dengan cara memberikan bobot pada setiap kata. Metode TF-IDF menggabungkan dua konsep yaitu frekuensi kemunculan suatu kata dalam suatu dokumen dan frekuensi kebalikan dari dokumen yang mengandung kata tersebut. Dalam menghitung bobot dengan menggunakan TF-IDF, terlebih dahulu dihitung nilai TF per kata dengan bobot setiap kata adalah 1. Sedangkan nilai IDF diformulasikan dalam persamaan berikut (Deolika, Kusrini, and Luthfi 2019) :

$$IDF\ (Word) = \log \frac{td}{tf} \quad (2.11)$$

Dimana IDF (*word*) merupakan nilai IDF dari setiap kata yang akan dicari, kemudian td merupakan jumlah dari keseluruhan dokumen yang ada, dan df merupakan jumlah kemunculan kata disetiap dokumen.

2.6 Metode SEMMA

SEMMA merupakan kepanjangan dari *Sample, Explore, Modify, Model* dan *Assess.* Didalam penelitian Shafique dijelaskan bahwa SEMMA merupakan sebuah metode *data mining* yang dikembangkan oleh SAS *Institute*. Metode ini membantu penelitian dalam memberikan solusi untuk tujuan maupun masalah bisnis. Maka dari itu Penelitian ini menggunakan metode SEMMA dengan tahapan sebagai berikut (Alizah et al. 2020) :



Gambar 2.3 Proses SEMMA

(Sumber : (Alizah et al. 2020))

2.6.1 *Sample*

Sample merupakan sebuah proses data mining yang dapat digunakan untuk mengumpulkan sampel yang digunakan untuk mencari data yang cukup besar dan dapat membentuk informasi yang penting dan signifikan, namun data tersebut dapat dimanipulasi dengan cepat. Tahap sample ini bersifat optional jadi untuk melakukan proses data mining tidak mewajibkan untuk melakukan sample untuk melakukan proses data tersebut.

2.6.2 *Explore*

Explore merupakan sebuah proses data mining yang dapat digunakan untuk mencari kumpulan data dan menjadi informasi yang terkait dengan tren dan anomaly yang tidak terduga yang dapat digunakan untuk mendapatkan pengertian dan ide. Jika eksplorasi visual tidak mengungkapkan tren yang jelas, maka dapat melakukan menjelajahi data melalui teknik statistik

termasuk analisis faktor, analisis korespondensi, dan pengelompokan untuk mendapatlan data yang jelas.

2.6.3 *Modify*

Didalam tahap ini terdapat beberapa pesiapan untuk data yang akan diolah, berikut tahapan persiapan tersebut

a. *Case Folding*

   Seluruh karakter huruf teks hasil *scrapping* dirubah menjadi huruf kecil.

b. Cleaning

   Dilakukan untuk menghilangkan URL atau tautan link website

c. Tokenize

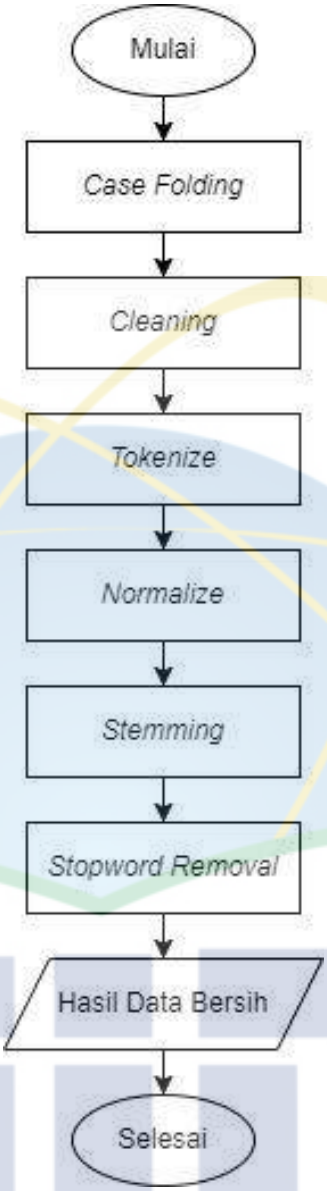
   Dilakukan untuk memecah sekumpulan kata (kalimat) menjadi kata yang memiliki arti tertentu (token). Hal ini bertujuan agar kata perkata dapat diolah lebih lanjut dalam proses-proses selanjutnya.

d. *Stemming*

   Tahap ini merupakan tahap dimana untuk memproses kata-kata untuk mengembalikan kata tersebut kedalam kata dasarnya. Dalam penelitian ini adalah bahasa Indonesia. Tahap ini bertujuan untuk membersihkan suatu kata dengan pengejaan yang kurang tepat.

e. *Stopword Removal*

   Tahap ini merupakan proses untuk melakukan filter terhadap kata-kata umum seperti “dan”, “atau”, dan lainnya, yang tidak diperlukan saat pemrosesan data.



Gambar 2.4 Alur Tahap *Modify*

(Sumber : (Alizah et al. 2020))

2.6.4 Model

Model merupakan sebuah proses data mining yang dapat digunakan untuk memodelkan data dengan menyediakan software untuk mencari kombinasi data yang memprediksi hasil terpercaya yang diinginkan secara otomatis. Data yang sudah dikombinasikan dapat digunakan untuk memprediksi hasil yang diinginkan.

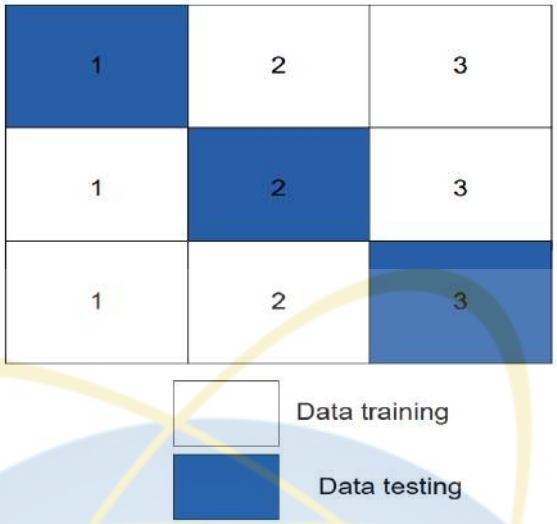
2.6.5 *Assess*

Assess merupakan sebuah proses data mining yang dapat digunakan untuk menilai data dengan mengevaluasi kegunaan dan keandalan penemuan dari data proses *data mining* dan memperkirakan seberapa baik kinerja tersebut.

2.7 *Cross Validation*

Cross Validation merupakan suatu aksi yang dilakukan untuk mencari tingkat akurasi dari tiap metode yang digunakan dengan membagi data berupa *training* dan *testing* berwujud *apply* model dan *performance* (Puspita and Widodo 2021). Jika menurut penelitian dari Tempola et al, pada tahun 2018, *cross-validation* dapat disebut dengan estimasi rotasi merupakan suatu teknik model validasi dalam menilai bagaimana hasil statistik analisis yang akan menyamaratakan kumpulan data independent (Tempola et al. 2018). Utamanya teknik ini digunakan untuk melakukan suatu prediksi model dan memprediksi keakuratan suatu model prediktif saat dijalankan dalam praktiknya.

Salah satu teknik atau jenis dari *cross validation* adalah k-*fold cross validation*. K-*fold cross validation* merupakan salah satu dari jenis *cross validation* yang mempunyai fungsi untuk menilai kinerja proses suatu metode algoritma dengan cara membagi sampel data secara *random* yang kemudian mengelompokkan data tersebut sebanyak nilai k. Setelah itu, salah satu kelompok k-*fold* tersebut dijadikan data uji sedangkan kelompok sisanya akan dijadikan sebagai data latih (Cahyanti, Rahmayani, and Husniar 2020).



Gambar 2.5 Model 3-*fold cross validation*

(Sumber: (Tempola et al. 2018))

Gambar diatas adalah penggunaan 3-*fold cross validation*. Masing-masing data akan dieksekusi sebanyak 3 kali dengan setiap subset data akan mempunyai kesempatan untuk menjadi data *testing* ataupun data *training*. Dengan diasumsikan nama setiap bagian D1, D2 dan D3 maka (Tempola et al. 2018):

1. Percobaan pertama dengan data D1 sebagai data *testing*, kemudian data D2 dan D3 adalah data *training*.
2. Percobaan kedua dengan data D2 sebagai data *testing*, kemudian data D1 dan D3 adalah data *training*.
3. Percobaan ketiga dengan data D1 sebagai data *testing*, kemudian data D2 dan D3 adalah data *training*.

Dengan membandingkan semua data uji yang diklasifikasi benar dengan banyaknya data uji merupakan cara untuk pengukuran *performance* klasifikasi. Berikut merupakan model yang digunakan dalam mengukur kinerja klasifikasi.

$$akurasi = \frac{\sum klasifikasi\ benar}{\sum data\ uji}\ x\ 100\%$$

(2.12)

Kemudian *standar deviation* atau simpangan baku selanjutnya akan dihitung. Simpangan bakumerupakan suatu ukuran penyebaran data yang menunjukkan jarak rata-rata nilai tengah ke suatu titik nilai. Jika simpangan baku yang dihasilkan semakin besar maka nilai tengah penyebarannya akan semakin besar, begitu juga sebaliknya. Simpangan baku memiliki tujuan untuk melihat jarak antara rata-rata akurasi dengan akurasi pada masing masing percobaan. Didalam penelitian Tempola yang mengutip brown berikut adalah cara menghitung simpangan baku (Tempola et al. 2018):

$$\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{N}} \qquad (2.13)$$

Dimana :

μ = mean

N = Banyaknya percobaan

X = Percobaan ke-i

i = indeks setiap percobaan

2.8 *Lexicon Based*

Lexicon Based adalah sebuah metode yang didasari oleh kamus yang bertujuan untuk mendapatkan bobot dari suatu kalimat didalam data set sehingga dapat diketahui sentimen dari kelas dataset tersebut (Herdhiyanto 2020). Metode *lexicon* bisa dilakukan atau dibuat dengan cara manual maupun secara otomatis dari *seed of*

word diperluas. Dengan digunakannya metode *lexicon based* data yang cocok digunakan seperti data kuesioner, data komentar youtube, data twitter atau semacam media sosial lainnya yang mengandung opini – opini dari pengguna maupun masyarakat luas. Ini dijelaskan dalam penelitian Matulatuwa pada tahun 2017, penelitian ini mengatakan metode *lexicon based* merupakan metode yang praktis, sederhana dan layak untuk menganalisis sentimen dari *platform* media sosial (Matulatuwa, Sediyono, and Iriani 2017).

Lexicon Based mempunyai beberapa kelebihan salah satunya adalah metode ini dapat melakukan pelabelan secara otomatis pada suatu kalimat sehingga terjadi penghematan waktu dalam proses pengolahan dataset yang berjumlah banyak atau besar (Azhar 2019). Kemudian dengan menggunakan metode ini, sentimen didalam dataset digunakan untuk menghindari kalimat bias dari opini pribadi seseorang.

Komponen terpenting didalam metode *lexicon based* adalah kamus. Kamus digunakan untuk menormalisasikan suatu kalimat dan mengekstrak kata kunci. Dengan contoh sebagai berikut :

a. Kata kunci positif : senang, bagus, keren, baik, cerdas.
b. Kata kunci negatif : bodoh, jelek, jahat, gagal, sulit, lemah.
c. Kata kunci negasi : tetapi, tidak, bukan, sebaiknya.

Orientasi sentimen kalimat ditentukan dengan menjumlahkan nilai orientasi semua kata sentimen dalam kalimat. Kata positif diberi nilai sentimen +1 dan kata negatif diberi nilai sentimen -1. Kata-kata negasi dan kata-kata yang berlawanan juga dipertimbangkan. Terdapat 4 langkah dalam menentukan suatu orientasi sentimen berdasarkan pendekatan *lexicon*, yaitu:

1. Identifikasi kata yang mengandung sentimen : Untuk setiap kalimat yang mengandung satu atau lebih kata sentimen, langkah ini memilih semua kata dan frasa dalam kalimat sentimen. Setiap kata positif diberi skor sentimen +1 dan setiap kata negatif diberi skor sentimen -1. Contoh "Barang ini kualitasnya kurang bagus [+1], tapi dayanya tahan lama [+1]". Dari contoh ini kata bagus bernilai +1 dan tahan lama juga bernilai +1 karena merupakan kata yang positif.
2. Pengubah sentimen (sentimen *shifter*) adalah kata dan frasa yang dapat mengubah arah sentimen. Ada beberapa jenis pengonversi kata negatif (*shifter* negasi), seperti tidak, tidak pernah, dan tidak ada yang merupakan jenis yang paling umum. Dengan demikian, kalimatnya menjadi “Kualitas dari barang ini tidak bagus [-1], tetapi kekuatannya tahan lama [+1]” karena kata negasinya “Tidak.”.
3. Agregasi: Pada langkah ini, fungsi agregasi opini diterapkan pada skor sentimen yang dihasilkan untuk menentukan arah akhir sentimen.

2.8.1 Kamus *Lexicon*

Dalam pendekatan analisis sentimen menggunakan *lexicon*, kamus merupakan komponen kunci dalam penggalian kata sentimen. Karena kebanyakan kamus seperti *WordNet* mengandung sinonim dan antonim untuk setiap kata, menggunakan kamus untuk mengumpulkan kata-kata sentimen adalah metode yang jelas. Jadi, singkatnya, teknik atau pendekatan ini adalah dengan menggunakan beberapa kata sentimen benih sebagai referensi dan mencocokkannya berdasarkan sinonim dan struktur antonimnya dalam kamus..

Secara khusus, metode ini berfungsi sebagai sekumpulan kecil kata sentimen dengan orientasi positif atau negatif yang diketahui yang kemudian dikumpulkan secara manual. Algoritma ini kemudian menghitung jumlah kata dengan mencari di *WordNet* atau kamus lain sesuai dengan sinonim dan antonim. Kata-kata yang ditemukan akan dimasukkan ke dalam daftar positif atau negatif. Proses berakhir ketika tidak ada kata baru yang ditemukan. Setelah proses selesai, langkah pemeriksaan digunakan untuk menghitung agregat positif atau negatif. (Bhonde, 2015)

Terdapat beberapa kamus yang akan digunakan dalam pendekatan *lexicon* seperti kamus *lexicon* positif, negatif, negasi, Kamus Besar Bahasa Indonesia (KBBI), kamus kata dasar, dan kamus *stopword*.

a. Kamus Positif

   Kamus positif digunakan untuk menyeleksi kata-kata yang termasuk dalam sentimen positif suatu kalimat atau query yang akan ditentukan sentimennya.

b. Kamus Negatif

   Kamus negatif digunakan untuk menyeleksi kata-kata yang termasuk dalam sentimen negatif suatu kalimat atau query yang akan ditentukan sentimennya.

c. Kamus negasi

   Kamus negasi digunakan untuk mendeteksi kalimat atau query yang memiliki sentimen yang telah ditentukan sebelumnya, baik positif maupun negatif, apakah sentimen tersebut diikuti dengan kata negasi.

Sentimen yang diikuti kata negasi akan mengalami perubahan nilai sentimen dari sebelumnya.

d. Kamus kata dasar dan KBBI

Kamus kata dasar dan Kamus Besar Bahasa Indonesia (KBBI) digunakan untuk melakukan proses *stemming* pada tahap natural *language processing*. *Stemming* adalah mengubah kata berimbuhan menjadi kata dasar. Dalam proses ini diperlukan kamus kata dasar dan KBBI sebagai pilihan kata yang tepat.

e. Kamus stopwords

Kamus *stopwords* digunakan untuk memilih kata-kata didalam dataset yang dianggap tidak penting. Proses ini dilakukan untuk mempercepat proses klasifikasi data.

2.9 YouTube

YouTube merupakan *platform* video *online* yang didirikan pada tahun 2005. Saat ini, YouTube dimiliki oleh Google yang mengakuisisi YouTube pada tahun 2006 ketika YouTube baru berjalan selama satu setengah tahun (Viertola 2018). Seluruh orang di dunia yang memiliki akses internet dapat mengakses, mengunggah, menonton, mengomentari dan membagikan video di YouTube tanpa bayaran sedikit pun. Oleh karena itu, YouTube telah menjadi salah satu platform terbesar di dunia dalam mengakses, mencari, menonton, berbagi, dan membuat konten video, di antara penggunaan khusus lainnya yang diberikan oleh penggunanya(Pires, Masanet, and Scolari 2021). Dengan lebih dari 1,9 juta pengguna bulanan (YouTube, 2019).

Platform media sosial seperti Twitter, Facebook, dan YouTube memiliki arsitektur, norma, dan budaya yang unik. Keunikan dari cara penggunaan YouTube dapat terlihat tidak hanya untuk hiburan tetapi juga untuk interaksi sosial dalam bentuk komentar, pencarian dan memberikan informasi yang menjadikan situs ini menarik dari perspektif penelitian (Khan 2017). YouTube yang dikenal dengan slogan "*Broadcast Yourself*" menempati peringkat pertama untuk situs web *sharing video* pada tahun 2011. Sementara itu, perkembangan YouTube di Indonesia dari *Head of Communications Consumer* YouTube Indonesia, Putri Silalahi mengatakan, jumlah penonton dan pembuat video *online* di YouTube tumbuh luar biasa di Indonesia. Durasi menonton orang di Indonesia meningkat 130% dari tahun 2014 ke 2015. Demikian juga jumlah konten yang diunggah meningkat 600% (Mellyaningsih 2016). Popularitas Youtube diprediksi akan terus meningkat dengan seiring bertambahnya pengguna. Pada pertengahan tahun 2017 Youtube mencatatkan jumlah penonton terdaftar sebanyak 1,5 miliar *user* (*logged-in monthly users*). Pada tahun 2019 lembaga riset pasar statista sudah memprediksikan bahwa pengguna (*users*) Youtube ini bisa mencapai hingga 1,8 miliar di tahun 2021 (Mujianto 2019).

2.9.1 YouTube API

Application Programming Interfaces (APIs), adalah cara untuk *libraries of code*, SDK, dan kerangka kerja tersedia untuk *developers* (Myers and Stylos 2016). Sudah banyak perusahaan yang menyediakan API sehingga orang lain dapat mengakses sedikit data dan layanan mereka. Misalnya, per Oktober 2017, programmableweb.com mencantumkan lebih dari 18.400 API untuk layanan Web. Mulai ada sejumlah komunitas panduan publik yang

berupaya untuk membantu pemrogram internal dan eksternal *developers* membuat API berkualitas tinggi (Murphy et al. 2017).

Youtube *Data API* merupakan penghubung antara fungsi yang biasanya dijalankan pada situs web YouTube ke dalam suatu situs web atau aplikasi yang dibuat oleh seorang *developer*. API juga mendukung metode untuk menyisipkan, memperbarui, atau menghapus banyak sumber daya (Google Developers 2023).

2.9.2 Video Youtube

Video berasal dari kata *vidi* atau *visum* yang artinya melihat atau mempunyai daya penglihatan. Munir dalam kutipan Yuanta di tahun 2019 mengatakan bahwa video adalah gambar yang bergerak yang disertai oleh suara. Media Video pun merupakan salah satu jenis media audio visual yang dapat menggambarkan objek yang bergerak dan disertai suara yang sesuai dengan gambar yang ada. Peran video ini pun adalah untuk penyaji informasi (Yuanta 2019).

Jika menurut jurnal yang disampaikan oleh Pambudi dan Afghohani pada tahun 2019, penelitian tersebut mengutip Sianipar didalam penelitiannya bahwa video adalah rangkaian dari beberapa *frame* (bingkai) gambar yang dijalankan secara cepat. *Frame* tersebut adalah rekaman suatu tahapan (sekuen) dari suatu gerakan. Jika kecepatan dari rangkaian gambar tersebut dijalankan dengan kecepatan di atas 20 *frame/second* maka mata kita tidak akan bisa menangkap perbedaan (titik jeda perpindahan) antar *frame* tersebut (Pambudi and Afghohani 2019).

Sedangkan pada penelitian dari Batubara dan Ariani yang mengutip Munir bahwa video mempunyai definisi sebagai suatu teknologi perekaman, pengolahan, pemindahan, penyimpanan, penangkapan, dan perekonstruksian urutan gambar yang diam dengan menyajikan beberapan adegan gerak secara elektronik sehingga video menjadi seperti gambar yang bergerak. Gabungan gambar tersebut disebut *frame*, kecepatan membaca *frame* disebut *frame rate* dengan satuan fps (*frame per second*), dan ukuran gambar disebut dengan resolusi (Batubara and Ariani 2016).

Video yang merupakan salah satu kemajuan teknologi telah membuat suatu kemajuan bagi manusia dan kebudayaan hingga memberikan pengaruh yang positif maupun negatif. Banyak orang yang sangat terbantu dengan adanya video dalam penerimaan informasi baik melalui video analog maupun video digital. Secara garis besar video terbagi 2, video analog merupakan video yang disimpan tidak didalam komputer seperti misalnya video televisi, video tape, dan film. Jenis ini memakai gelombang analog. Sedangkan video digital adalah video yang diproduksi oleh industri komputer dengan sederet bilangan biner (Batubara and Ariani 2016) seperti misalnya media sosial video, dan YouTube termasuk didalamnya.

Menurut penelitian atau jurnal dari Pambudi dan Afghohani yang mengutip Anwari mengenai YouTube adalah sebuah situs web video *sharing* (berbagi video) yang popular. Situs YouTube adalah video *sharing* yang terbesar yang pernah ada (Pambudi and Afghohani 2019). Dengan penjelasan tersebut video YouTube merupakan kumpulan gambar tidak bergerak dengan menyajikan adegan bergerak secara elektronik, kumpulan gambar (*frame*)

tersebut disimpan dan diproduksi oleh industri komputer (video *digital*) yang diletakkan pada *platform* media sosial Youtube.

2.9.3 YouTube Content *Creator*

Meningkatnya minat masyarakat Indonesia akan penggunaan YouTube, hal tersebut memicu munculnya *content creator* yang memilih YouTube sebagai *platform* media sosial untuk berkarya. Dengan begitu, para *content creator* bersaing untuk menghasilkan suatu karya yang dapat dinikmati oleh masyarakat khususnya oleh pengguna YouTube (Christianto, Andjarwirawan, and Tjondrowiguno 2020). *Content creator* atau yang dapat disebut *influencer* merupakan seseorang yang membuat suatu *content* dimana orang tersebut mempunyai banyak pengikut di media sosialnya. *Content creator* biasanya membangun suatu ikatan atau *engagement* dengan para pengikutnya menggunakan cara seperti membagikan konten yang menghibur, menginspirasi, memberikan pengetahuan baru dan informasi yang bisa menyatukan mereka dengan pengikutnya di media sosial (Larasati et al. 2021).

Salah satu *content creator* YouTube Indonesia adalah Raymond Chin. Raymond Chin merupakan seorang pembuat konten video di YouTube yang membahas mengenai topik bisnis dan ekonomi. Bergabung di YouTube sejak 23 Januari 2014, Raymond Chin mempunyai jumlah *subscriber* atau bisa dikatakan pengikut kanal YouTubenya sebanyak 811.000 pengguna. Sebagai *CEO & Founder* Ternak Uang, ia sangat memperhatikan isu isu ekonomi dan bisnis yang terjadi di Indonesia. Baik untuk marketing produknya maupun memberikan pengetahuan dan informasi pada masyarakat luas. Salah satu

video yang ia buat adalah video yang berjudul "2023: Menuju KEHANCURAN DUNIA" yang bertopik mengenai resesi yang akan terjadi di tahun 2023.

Raymond Chin dan kebanyakan *content creator* lain biasanya mengecek komentar video Youtube mereka guna untuk meningkatkan kualitas video mereka baik dari segi *editing* maupun isi dari video tersebut. Biasanya dilakukan secara manual dengan membaca satu per satu komentar pada video mereka. Realitanya data komentar video yang ada begitu banyak sehingga pengecekan secara manual tidak efektif karena akan membutuhkan waktu yang lama serta menggunakan sumber daya manusia yang intensif. Dengan menggunakan web *scrapping* dan dilanjutkan dengan analisis sentimen, maka hal tersebut dapat dilakukan dengan efektif.

2.10 Web *Scraping*

Suatu teknik untuk mendapatkan informasi secara otomatis dan tanpa harus melakukannya secara manual dari website, dalam hal ini website Youtube dapat disebut dengan web *scraping*. Fokus pada web *scraping* adalah untuk mendapatkan data dan infornasi berupa teks atau tautan dengan proses pengambilan dan ekstraksi untuk mengambil data tertentu dari suatu ahalaman agar dapat digunakan Kembali oleh sistem lain maupun dianalisis lebih lanjut (Fikri, Handayanto, and Irwan 2022). Terdapat beberapa istilah untuk metode ini seperti *screen scraping*, *web harvesting*, ataupun metode lain yang sejenis.

Secara teori, web scraping merupakan suatu cara pengumpulan data melalui metode yang berbeda dengan menggunakan *Application Programming Interface*

(API). Penulisan kode program biasanya cara awal yang dilakukan. Ini digunakan sebagai otomatisasi *query* untuk meminta data terhadap server. Data hasil permintaan tersebut dapat selanjutnya dilakukan ekstraksi untuk mendapatkan hasil yang dicari. Langkah web scraping dapat dilihat pada gambar berikut (Sahria 2020) :

Gambar 2.6 Alur Web *Scraping*

(Sumber : (Sahria 2020))

Terdapat manfaat dari web *scraping* seperti misalnya ialah agar informasi yang ditambang lebih terfokuskan sehingga dalam pencarian sesuatu dapat dimudahkan (A. Yani, Pratiwi, and Muhardi 2019). Biasanya data yang didapatkan berupa halaman web dokumen HTML dengan memilih bagian tertentu yang kemudian dapat ditransformasi dari bentuk yang tidak terstruktur dalam format HTML menjadi suatu data yang terstruktur yang selanjutnya dapat disimpan dalam format data tertentu (Fikri et al. 2022).

2.11 Resesi

Resesi adalah kondisi dimana pertumbuhan ekonomi riil tumbuh negatif atau dengan kata lain terjadi penurunan produk domestik bruto selama dua kuartal berturut-turut dalam satu tahun (Blandina et al. 2020). Resesi ditandai dengan melemahnya

ekonomi global dan akan mempengaruhi ekonomi domestik negara-negara di seluruh dunia. Kemungkinan suatu negara mengalami resesi semakin kuat jika perekonomian negara tersebut bergantung pada ekonomi global (Miraza 2019).

Gambar 2.7 Infografis Indonesia Masuk Resesi Ekonomi

(Sumber: (Sunarmin and Junaidi 2021))

Seiring berkembangnya kasus pandemi COVID-19 tahun 2021, pasar berfluktuasi lebih ke arah negatif. Tidak hanya itu, perlambatan ekonomi global khususnya kegiatan ekspor Indonesia ke China juga berdampak signifikan terhadap perekonomian Indonesia. Hal ini didasarkan pada analisis sensitivitas yang menjelaskan bahwa perlambatan ekonomi global saat ini berdampak besar terhadap pertumbuhan ekonomi Indonesia. Masuknya Indonesia ke jurang resesi juga sudah diprediksi jauh-jauh hari. Pada September 2020, Menteri Keuangan Sri Mulyani memperkirakan pertumbuhan ekonomi Indonesia triwulan III 2020 mencapai -2,9 hingga -1,0 % (Sunarmin and Junaidi 2021).

2.12 Rapid Miner

RapidMiner adalah software/perangkat lunak untuk mengolah data. Menggunakan prinsip dan algoritma penambangan data (*data mining*), RapidMiner mengekstraksi pola dari kumpulan data besar dengan menggabungkan metode statistik, kecerdasan buatan, dan basis data (Rahmat et al. 2017). RapidMiner perangkat lunak yang dapat diakses oleh siapa saja dan bersifat terbuka (open source). RapidMiner digunakan sebagai solusi untuk menganalisis pemrosesan data. Pada RapidMiner sendiri, digunakan berbagai macam teknik seperti teknik deskriptif dan prediktif dalam memberikan wawasan kepada pengguna sehingga dapat membuat keputusan yang paling baik (Sari et al. 2020).

RapidMiner merupakan perangkat lunak yang berdiri sendiri untuk analisis data dan sebagai mesin penambangan data yang dapat diintegrasikan ke dalam produknya. RapidMiner ditulis menggunakan bahasa Java sehingga dapat bekerja di semua sistem operasi (Nabila, Isnain, and Abidin 2021). RapidMiner memudahkan pengguna untuk melakukan perhitungan data besar menggunakan operator. Operator ini digunakan untuk memodifikasi data. Data terhubung ke node pada operator kemudian peneliti hanya perlu menghubungkannya ke node hasil untuk melihat hasilnya. Hasil yang ditunjukkan oleh RapidMiner juga dapat ditampilkan secara visual dengan grafik (Rahmat et al. 2017).

2.13 Python

Python adalah bahasa pemrograman yang kuat dan mudah dipelajari. Python memiliki struktur data tingkat tinggi yang efisien dan pendekatan yang sederhana namun efektif untuk pemrograman berorientasi objek (Rossum and Drake 2020). Situs

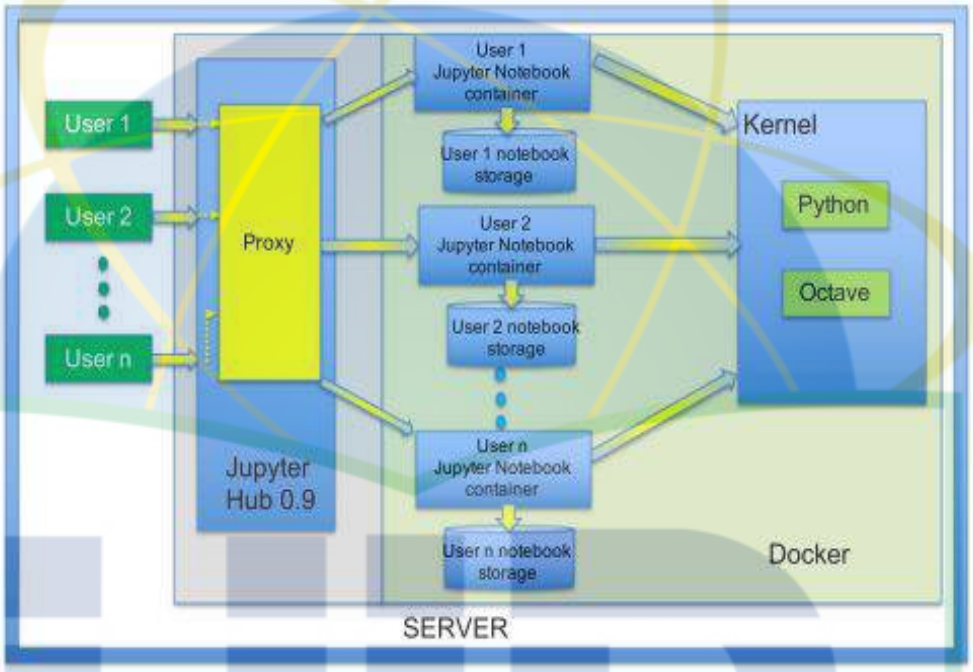
web python www.python.org dapat didistribusikan secara bebas dan juga menyimpan distribusi dan petunjuk ke banyak modul python pihak ketiga secara gratis. Python sering kali digunakan dalam pengembangan berbagai macam perangkat lunak, misalnya *product customization, internet scripting, numberic programming, systems programming,* dan sebagainya (Saputra and Aji 2018). Python juga dapat dikatakan sebagai bahasa pemrograman interpretatif multiguna yang dapat melakukan eksekusi sejumlah instruksi multi guna secara langsung (interpretatif) dengan metode Object Oriented Programming dan menggunakan semantik dinamis yang terfokus pada tingkat keterbacaan suatu *syntax* (Syaefulloh 2020).

Menurut salah satu penelitian, yaitu penelitian dari Fikri, Handayanto, dan Irwan menyatakan bahwa Python dapat diklaim sebagai bahasa yang menggabungkan kemampuan dan kapabilitas dengan sintaksis kode yang jelas kemudian dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif (Fikri et al. 2022). Saat ini, mungkin antarmuka yang paling ramah pengguna untuk Python adalah Notebook Jupyter, yang menyediakan antarmuka interaktif untuk bahasa pemrograman Python dan cocok untuk sebagian besar pekerjaan analitik data.

2.14 Jupyter *Notebook*

Jupyter adalah singkatan dari tiga bahasa pemrograman, yaitu Julia (Ju), Python (Py), dan *notebook* R. Jupyter adalah aplikasi web gratis yang paling banyak digunakan oleh *data scientist* saat ini. Aplikasi ini digunakan untuk membuat dan berbagi dokumen yang memiliki kode, hasil perhitungan, visualisasi dan teks(Susanti and Walid 2022). Saat ini, perangkat lunak *open source* Jupyter-*notebook* semakin banyak digunakan dalam pendidikan, terutama di pendidikan tinggi. Selain itu, karena

kesederhanaannya, *support comunity* yang aktif, dan ketersediaan pustaka numerik dan plotting seperti numpy, os, time, IPython, matematika, scipy, dan matplotlib, membuat *notebook* Python Jupyter telah menjadi salah satu bahasa pemrograman paling sering digunakan (Zúñiga-López and Avilés-Cruz 2020). Berikut merupakan skema umum dari sever *notebook* Jupyter



Gambar 2.8 skema umum server *Jupyter Notebook*

(Sumber: (Zúñiga-López and Avilés-Cruz 2020))

Dalam penelitian ini peneliti akan menggunakan Jupyter *notebook* untuk menjalankan perintah dalam bahasa pemrograman Python untuk menganalisa data teks. Jupyter notebook lebih memudahkan pengguna atau dapat dikatakan bahwa Jupyter *Notebook user friendly.*

2.15 Penelitian Sejenis

Penelitian sejenis atau literature sejenis adalah literature review dimana penulis mengumpulkan berbagai penelitian sebelumnya yang berkaitan atau memiliki topik yang mirip dengan penelitian terkait, sehingga penulis dapat membuat perbandingan dan penelitian terdahulu juga dapat dijadikan referensi oleh penulis. Riset serupa datang dari berbagai tesis dan jurnal yang memiliki topik analisis sentimen menggunakan metode K-NN dan NBC sebagai metode utama pada bidang tertentu. Berikut ini adalah kumpulan literatur serupa.

Tabel 2.1 Jurnal Penelitian Sejenis

No	Peneliti	Tahun	Judul Penelitian
1	Harpizon et al	**2022**	Analisis Sentimen Komentar Di YouTube Tentang Ceramah Ustadz Abdul Somad Menggunakan Algoritma Naïve Bayes
Objek : Menggunakan 1000 komentar dari 10 video yang ada di Youtube mengenai Ustadz Abdul Somad. **Metode**: Naïve Bayes **Tujuan** : Menganalisis sentimen masyarakat terhadap Ustadz Abdul Somad melalui komentar youtube menggunakan algoritma Naïve Bayes. **Hasil** : Didapatkan sebanyak 67% berkomentar positif, 27% berkomentar netral dan 6% berkomentar negatif. Berdasarkan pengujian didapatkan akurasi sebesar 87%, presisi 91% dan recall 97%. Berdasarkan pengujian tersebut dapat disimpulkan bahwa penelitian ini dapat digunakan untuk hasil sentimen dengan cepat di Youtube.			
2	Fatmasari et al.	**2022**	Analisis Sentimen Dalam Pengkategorian Komentar Youtube Terhadap Layanan Akademik dan Non-Akademik Universitas Terbuka Untuk Prediksi Kepuasan

Objek : Total dataset awal 7776 data dan setelah dilakukan cleansing dan preprocessing berjumlah 6920 data. **Metode**: Decision Tree (DT), Support Vector Machine (SVM), Naïve Bayes (NB) dan Random Forest (RF). **Tujuan** : Menganalisa prediksi kepuasan layanan dari beberapa kategori sebagai tolak ukurnya. **Hasil** : menghasilkan akurasi tertinggi pada kategori modul dengan nilai akurasi 100% dengan menggunakan algoritma DT, kategori Non Akademik dengan nilai akurasi 99,90% dengan menerapkan algoritma DT. kategori Ujian dengan nilai akurasi 99,70% dengan menerapkan algoritma RF, kategori Pengajar dengan nilai akurasi 99,42% dengan menerapkan algoritma DT, kategori modul dengan nilai akurasi 99,37% dengan menerapkan algoritma DT, kategori others dengan nilai akurasi 96,58% dengan menerapkan algoritma DT, dan kategori tutorial dengan nilai akurasi 92,40% dengan menerapkan algoritma SVM.			
3	Zulqornain et al.	2021	Analisis Sentimen Tanggapan Masyarakat Aplikasi Tiktok Menggunakan Metode Naïve Bayes dan Categorial Propotional Difference (CPD)
Objek : Ulasan yang diambil dari Google Play Store mulai dari rating 1 sampai dengan rating 5 dengan total 1000 data. **Metode**: Naïve Bayes dan Categorial Propotional Difference. **Tujuan** : Melakukan analisis sentimen pada ulasan aplikasi TikTok untuk membantu orang tua dalam pemilihan aplikasi untuk anaknya. **Hasil** : Hasil maksimal yang didapatkan dengan menggunakan 100% term yang digunakan untuk pengujian dengan nilai accuracy sebesar 0,729947 , nilai precission sebesar 0,746854, nilai recall sebesar 0,926118, dan nilai f-measure 0,824511.			
4	Puspita dan Widodo	2021	Perbandingan Metode KNN, Decision Tree, dan Naïve Bayes Terhadap Analisis Sentimen Pengguna Layanan BPJS
Objek : Pengguna BPJS pada media sosial Twitter sebanyak 1000 data. **Metode**: KNN, Decision Tree, dan Naïve Bayes.			

Tujuan : Untuk mengetahui tingkat, akurasi, dari, tiga metode, berbeda diantaranya KNN, Decision Tree dan Naïve, Bayes, dengan menggunakan tools RapidMiner. **Hasil** : Menunjukan bahwa analisis sentimen data Twitter terhadap layanan BPJS dengan menggunakan metode KNN mencapai tingkat akurasi 95.58% dengan class precision untuk pred. negative adalah 45.00%, pred. positive adalah 0.00%, dan pred. neutral adalah 96.83%. Lalu pada metode Decision Tree tingkat akurasinya mencapai 96.13% dengan class precision untuk pred. negative adalah 55.00%, pred. positive adalah 0.00%, dan pred. neutral adalah 97.28%. Dan yang terakhir adalah metode Naïve Bayes yang mencapai akurasi 89.14% dengan class precision untuk pred. negative adalah 16.67%, pred. positive adalah 1.64%, dan pred. neutral adalah 98.40%.			
5	Samsir et al.	2021	Analisis Sentimen Pembelajaran Daring Pada Twitter di Masa Pandemi COVID-19 Menggunakan Metode Naïve Bayes
Objek : Tweet dalam bahasa Indonesia dengan kata kunci ‘pembelajaran daring’, ‘kuliah’, ‘belajar’, ‘online’, ‘daring’, dan tagar #BelajarDariRumah dengan jumlah dataset sebanyak 12,906 tweet pada minggu pertama bulan November 2020 **Metode**: Naïve Bayes Classifier (NBC) **Tujuan** : Menganalisis opini publik terhadap pembelajaran daring pada masa pandemi COVID-19 di Indonesia pada awal November 2020. **Hasil** : Analisis sentimen terhadap calon gubernur DKI Jakarta 2017. Akurasi tertinggi didapat saat menggunakan metode klasifikasi Naïve Bayes Classifier (NBC), dengan nilai rata-rata akurasi mencapai 95%, nilai presisi 95%, nilai recall 95% nilai TP rate 96,8% dan nilai TN rate 84,6%.			
6	Nurjanah, Perdana, dan Fauzi	2017	Analisis Sentimen Terhadap Tayangan Televisi Berdasarkan Opini Masyarakat pada Media Sosial Twitter menggunakan Metode K-Nearest Neighbor dan Pembobotan Jumlah Retweet
Objek : Opini masyarakat terhadap tayangan televisi pada twitter sejumlah 400.			

Metode: KNN (K-Nearest Neighbor). **Tujuan** : Menjadi acuan serta pertimbangan bagi pemirsa tayangan televisi dan dalam memilih tayangan yang banyak disukai oleh masyarakat umum. **Hasil** : Akurasi menggunakan pembobotan tekstual diperoleh 82,50%, menggunakan pembobotan non-tekstual 60%, dan menggunakan penggabungan keduanya 83,33% dengan nilai k=3 dan konstanta perkalian yang tepat α=0,8 dan β=0,2.			
7	Ghulam Asrofi Buntoro	2017	Analisis Sentimen Calon Gubernur DKI Jakarta 2017 Di Twitter
Objek : Tweet dalam bahasa Indonesia dengan kata kunci AHY, Ahok, Anies, dengan jumlah dataset sebanyak 300 tweet **Metode**: Naïve Bayes Classifier (NBC) dan Support Vector Machine (SVM) **Tujuan** : Membantu untuk melakukan riset atas opini masyarakat yang mengandung sentimen positif, netral atau negatif **Hasil** : Analisis sentimen terhadap calon gubernur DKI Jakarta 2017. Akurasi tertinggi didapat saat menggunakan metode klasifikasi Naïve Bayes Classifier (NBC), dengan nilai rata-rata akurasi mencapai 95%, nilai presisi 95%, nilai recall 95% nilai TP rate 96,8% dan nilai TN rate 84,6%.			

BAB 3

METODOLOGI PENELITIAN

3.1 Metodologi Pengumpulan Data

3.1.1 Observasi

Observasi dilakukan untuk pengumpulan informasi dan data dengan melakukan pengamatan dan tinjauan secara langsung terhadap objek yang akan diteliti. Objek tersebut adalah tanggapan pengguna YouTube pada kolom komentar video Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA" dimana video tersebut mengenai prediksi resesi pada tahun 2023. Observasi ini dilakukan pada tanggal 06 Februari 2023 pada jam 12.10 WIB. Dengan jumlah komentar dalam jangka waktu 3 Oktober 2022 hingga waktu tersebut sebanyak 11.524 komentar.

3.1.2 Studi Pustaka

Studi Pustaka didalam penelitian ini dilakukan untuk menunjang pelaksanaan penelitian ini dengan cara membaca, memahami dan mempelajari teori teori yang terkait dengan pembahasan penelitian ini yang sudah dilakukan sebelumnya. Penelitian ini juga menggunakan beberapa referensi seperti dari penelitian sejenis, baik berbentuk buku fisik ataupun elektronik (*e-book*), jurnal artikel, skripsi maupun tesis dalam upaya untuk pemecahan masalah.

3.1.3 Studi Literatur

Studi literatur didalam penelitian ini dilakukan untuk mencari, memahami dan mempelajari penelitian-penelitian serupa dengan penelitian ini. Bertujuan untuk menjadikan acuan dan perbandingan dengan penelitian ini. Berikut merupakan tabel literatur sejenis yang digunakan :

Tabel 3.1 Studi Literatur

No	Peneliti	Tahun	Judul
1	Nur Adinda Salsabila	2022	ANALISIS SENTIMEN PADA MEDIA SOSIAL TWITTER TERHADAP TOKOH GUS DUR MENGGUNAKAN METODE *NAÏVE BAYES* DAN *SUPPORT VECTOR MACHINE* (SVM)
2	M. Rizqy Ariel Giffari	2022	ANALISIS SENTIMEN BERBASIS ASPEK PADA ULASAN APLIKASI TANGERANG *LIVE* MENGGUNAKAN *LATENT DIRICHLET ALLOCATION* DAN *NAÏVE BAYES*
3	Amalia K	2022	ANALISIS SENTIMEN DAN DETEKSI EMOSI DENGAN PENDEKATAN *LEXICON* PADA JUDUL BERITA MEDIA *ONLINE* MENGENAI COVID-19 DI INDONESIA
4	Ahmad Fatihin	2022	ANALISIS SENTIMEN TERHADAP ULASAN APLIKASI *MOBILE* MENGGUNAKAN METODE *SUPPORT VECTOR*

			MACHINE (SVM) DAN PENDEKATAN *LEXICON BASED*
5	Raffie Rizky F	2022	ANALISIS SENTIMEN DAMPAK EKONOMI MASYARAKAT INDONESIA AKIBAT PANDEMI COVID-19 PADA MEDIA SOSIAL TWITTER MENGGUNAKAN METODE *NAÏVE BAYES CLASSIFIER, SUPPORT VECTOR MACHINE* DAN *LEXICON BASED*

3.2 Metode SEMMA

Penelitian ini mengacu kepada metode SEMMA *Data Mining Process* untuk metodologi penelitian. Fokus didalam metode SEMMA ini adalah modifikasi, pemodelan, dan penambangan data yang dirancang untuk membantu pengguna SAS *enterprise miner*. Metode SEMMA terdiri dari *sample, explore, modify, model and asses.*

3.2.1 *Sample*

Pada tahap ini dilakukan pengumpulan data terkait tema dari penelitian terdahulu sebagai landasan penelitian. Dilakukan juga pengumpulan dataset yang bersumber dari komentar video YouTube Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA" dengan teknik *scraping* komentar yang dilakukan pada Februari 2023. Komentar yang didapat berjumlah 11.524 komentar dengan menggunakan *App Script* yang memanfaatkan fungsi YouTube API komentar-komentar tersebut disimpan didalam file csv.

Langkah yang akan dilakukan oleh peneliti adalah melakukan *login* atau masuk akun google untuk mengakses fitur-fitur yang disediakan oleh google seperti YouTube API, app script serta aplikasi-aplikasi pendukung lainnya. Kemudian, dari fitur-fitur tersebut peneliti dapat menggunakannya untuk proses *crawling data* dengan menggunakan Bahasa pemrograman python.

3.2.2 *Explore*

Pada tahap ini peneliti mendeskripsikan data yang didapat dari hasil crawling / *scraping* data komentar video YouTube Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA". Kemudian dilakukan tahap visualisasi data yaitu data yang didapat dibuat grafik berdasarkan waktu untuk menampilkan informasi dari data secara visual.

3.2.3 *Modify*

Pada tahap Modify di penelitian ini adalah tahapan untuk persiapan data atau pre-processing. Tahapan yang akan dilakukan terdiri dari *case folding, cleaning, tokenize, normalize, stemming, dan stopword removal*.

3.2.4 *Model*

Pada tahap ini dilakukan pelabelan data berdasarkan kelasnya untuk menentukan opini positif atau negatif menggunakan bahasa pemrograman python. Setelah itu, data set berupa data latih dan data uji akan diolah berdasarkan kedua metode yaitu KNN dan Naïve bayes.

3.2.5 *Assess*

Pada tahap ini dilakukan evaluasi dari model. Pada metode K-Nearest Neighbor dan metode Naïve Bayes Classifier digunakan 10-folds cross validation untuk mengetahui akurasi, presisi, recall dan f1-score dari model yang digunakan.

3.3 Perangkat Penelitian

Pada penelitian ini, peneliti menggunakan perangkat keras (*hardware*) dan perangkat lunak (*software*) dengan spesifikasi sebagai berikut :

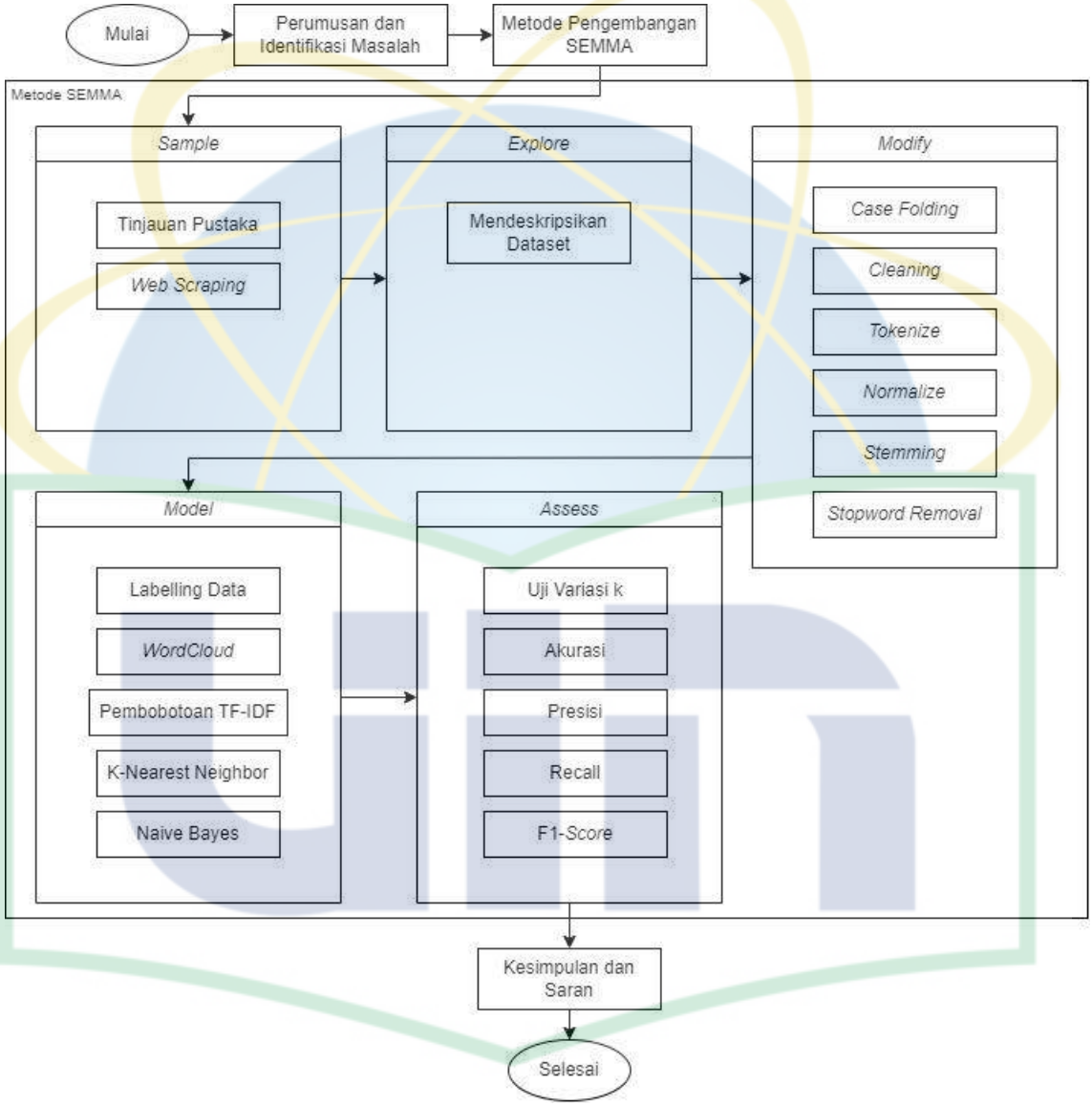
Tabel 3.2 Perangkat Penelitian Peneliti

Hardware	Laptop HP-D14MGQ2	AMD A8-740 APU 4CPUs, 2.2Ghz
		AMD Radeon R5 Graphics
		8 GB RAM
		512 GB HDD dan 256 GB SSD
Software	*Operating System*	Windows 10 Pro 64-bit
	Tools	Rapid Miner dan Jupyter *Notebook*
	Bahasa Pemrograman	Python

3.4 Alur Penelitian

Alur penelitian merupakan alur yang dilakukan untuk melakukan penelitian mulai dari awal hingga selesai. Terdapat beberapa tahapan pada alur penelitian pada penelitian ini dimulai dari penulis mengidentifikasi dan merumuskan masalah yang ada, kemudian dilanjut dengan pemilihan metode penelitian yaitu metode SEMMA yang memiliki 5 tahapan terdiri atas *sample, explore, modify, model*, dan *assess*.

Metode SEMMA tersebut didalamnya menggunakan algoritma Naïve Bayes dan KNN dalam pengolahan data. Setelah melakukan tahapan metode SEMMA yang sudah dijelaskan sebelumnya. Tahap terakhir adalah pembahasan mengenai hasil yang didapat dan kesimpulan serta saran dari penelitian. Berikut adalah alur penelitian



Gambar 3.1 Alur Penelitian

BAB 4

HASIL DAN PEMBAHASAN

4.1 *Sample*

4.1.1 Penelitian Sejenis

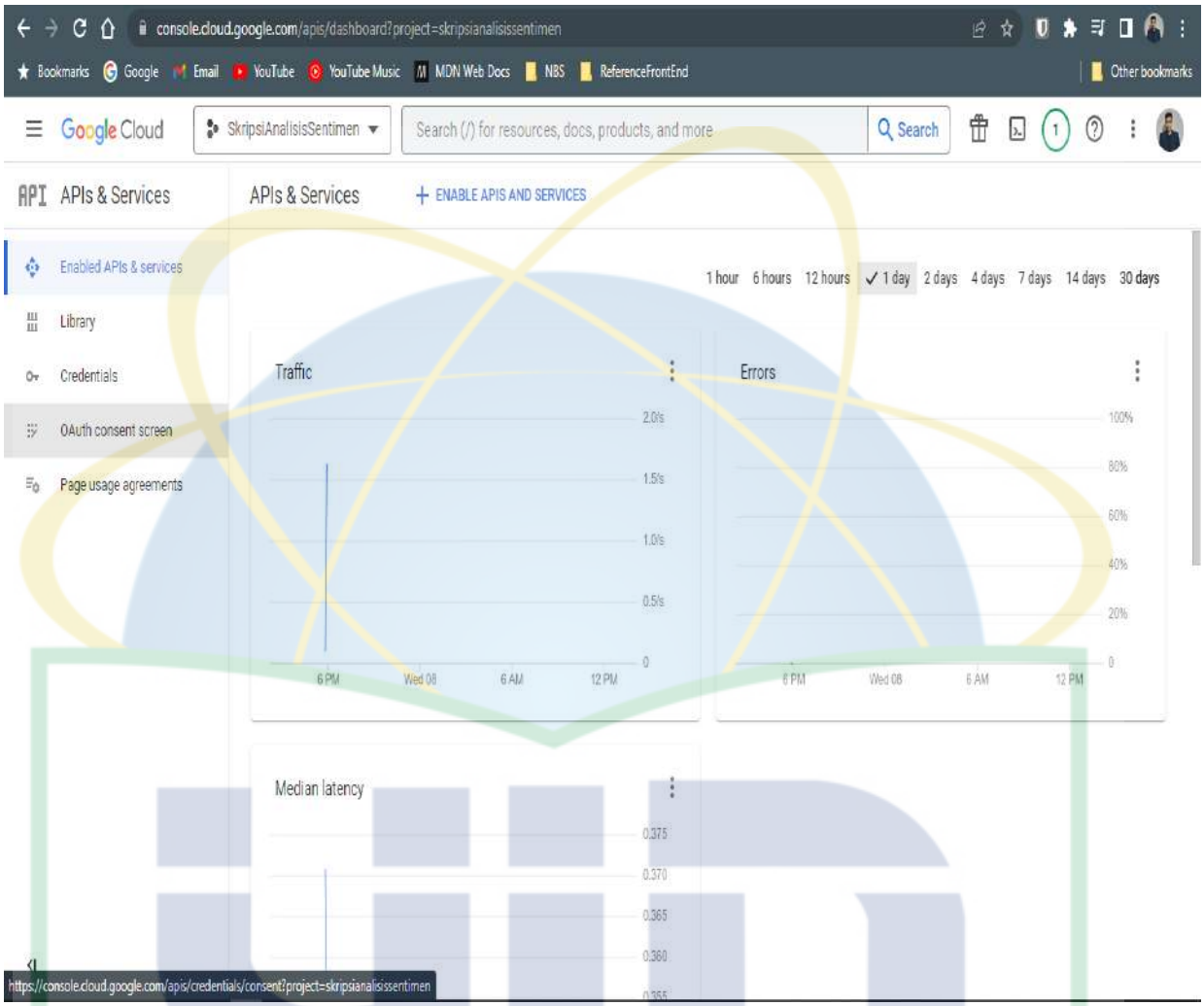
Penelitian sejenis merupakan tahap yang dilakukan terhadap materi analisis sentimen dengan metode Naïve Bayes dan atau metode K-*Nearest Neighbor* (KNN) dari jurnal dan penelitian yang telah dijelaskan sebelumnya pada Tabel 2.1

4.1.2 *Scraping Data*

Penelitian ini menggunakan sumber dataset dari komentar video YouTube Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA". Peneliti melakukan *scraping data* dengan menggunakan bahasa pemrograman python dan diterapkan dengan *notebook jupyter* di google colab. Berikut merupakan langkah – langkah yang dilakukan :

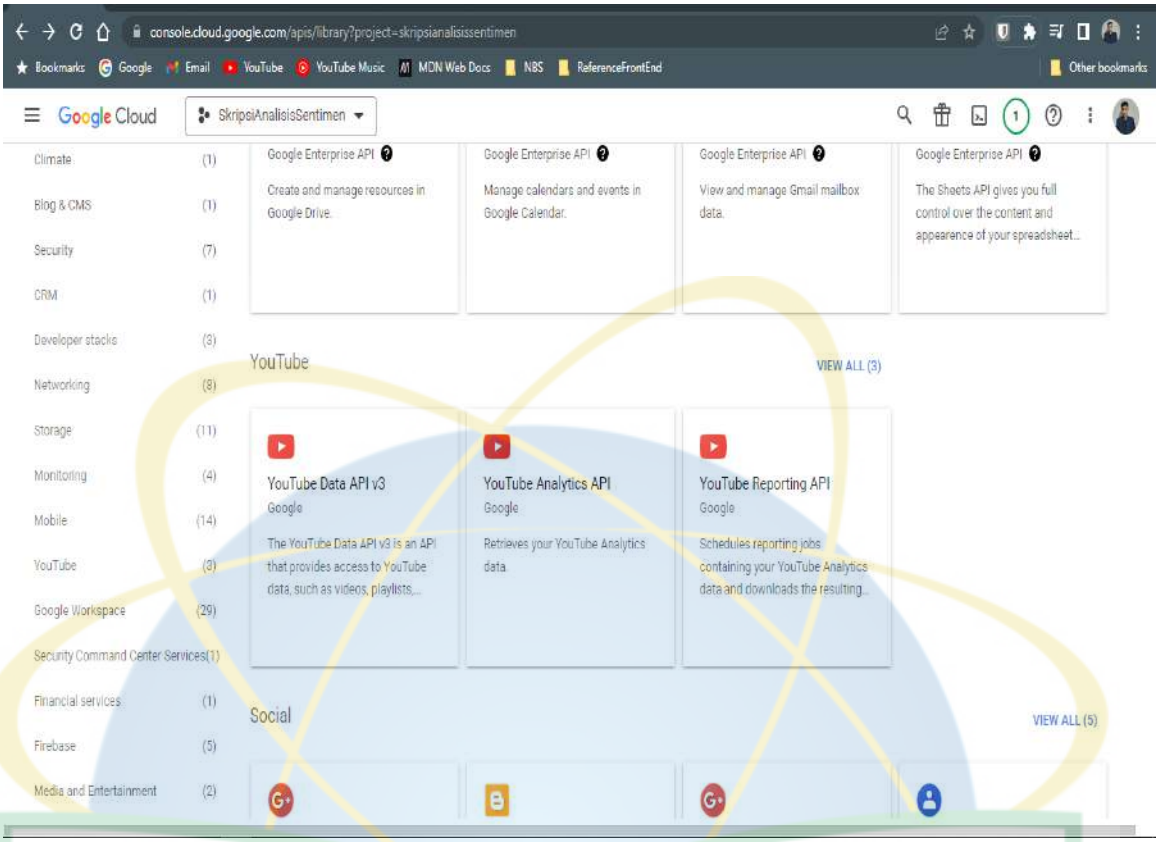
1. Mengunjungi https://accounts.google.com untuk melakukan login pada akun google yang dipunya. Jika belum mempunyai akun google maka registrasikan diri untuk mempunyai akun google dan masuk kedalam akun google tersebut.
2. Mengaktifkan layanan YouTube Data API dan mendapatkan API Key yang nantinya API tersebut digunakan dalam proses pemrograman dalam *scraping* dataset. Pengaktifan tersebut dengan cara mengunjungi

https://console.cloud.google.com/apis/dashboard dengan akun google yang sudah dibuat sebelumnya.



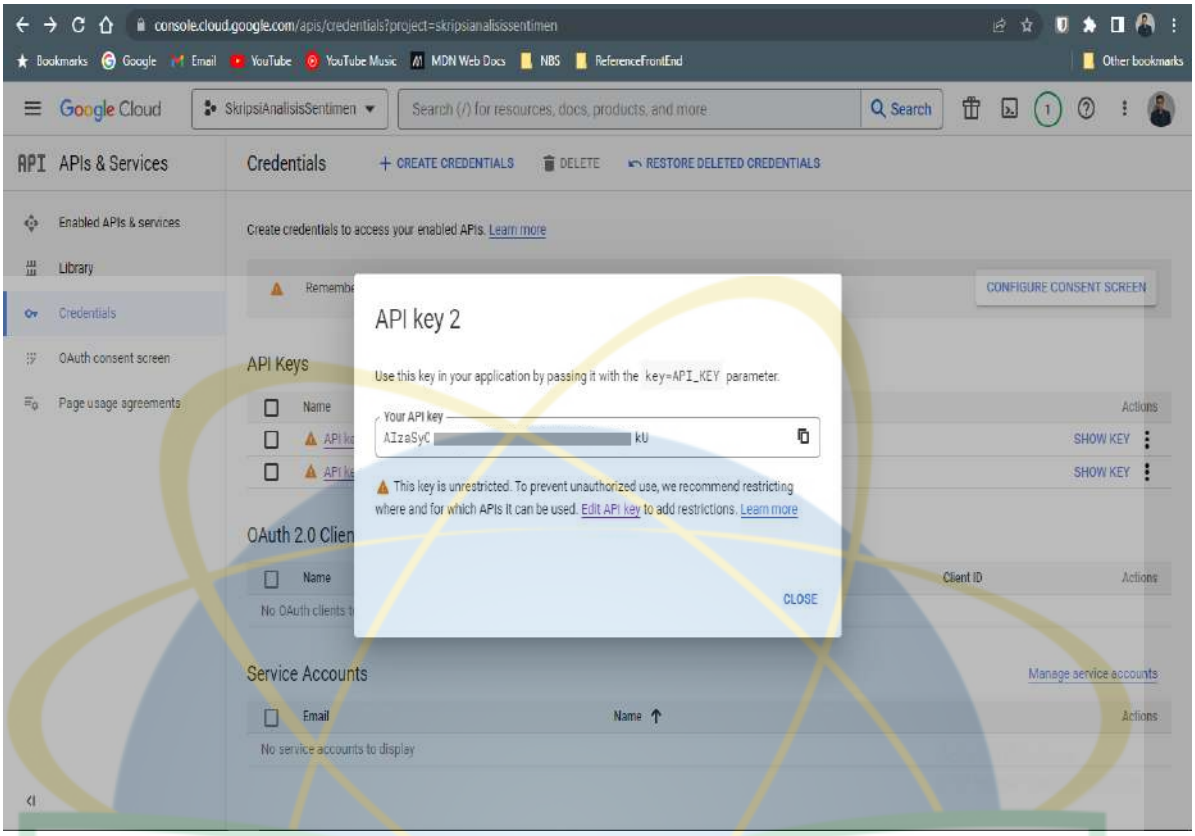
Gambar 4.1 *Dashboard* Google *Developer*

3. Sebelum mendapatkan API YouTube, peneliti harus membuat *enable API* untuk Youtube dengan cara menekan tombol "+ ENABLE APIS AND SERVICE" yang selanjutnya diarahkan kepada *library API* google dan pilih YouTube Data API v3 kemudian *enable*.



Gambar 4.2 *Library* API YouTube

4. Jika YouTube Data API v3 sudah *enable*, kemudian kembali ke halaman *dashboard* lalu masuk ke halaman *Credentials* untuk mendapatkan API Key. Selanjutnya tekan tombol "+ CREATE CREDENTIALS" lalu kemudian klik "API KEY" lalu akan muncul API Key yang akan digunakan selanjutnya pada proses pemrograman *scraping* komentar YouTube.



Gambar 4.3 ***Pop-up API Key*** **YouTube Data**

5. Selanjutnya proses pemrograman dengan menggunakan bahasa pemrograman python pada *notebook jupyter* di googlecolab dengan Langkah langkah sebagai berikut :

   a. Import *library*

```
import pandas as pd
from googleapiclient.discovery import build
```

Gambar 4.4 Kode ***Import Library***

   b. Kemudian membuat fungsi *scraping* komentar dengan memasukkan API Key yang sebelumnya sudah didapatkan dari pembuatan *credential* pada google developer. Dari pemrograman tersebut maka akan muncul komentar – komentar atau hasil data kometar dari video YouTube.

```python
#import library

import pandas as pd
from googleapiclient.discovery import build

def video_comments(video_id):
    # empty list for storing reply
    replies = []
    # creating youtube resource object
    youtube = build('youtube', 'v3', developerKey=api_key)
    # retrieve youtube video results
    video_response = youtube.commentThreads().list(part='snippet,replies', videoId=video_id).execute()
    # iterate video response
    while video_response:
        # extracting required info
        # from each result object
        for item in video_response['items']:
            # Extracting comments ()
            published = item['snippet']['topLevelComment']['snippet']['publishedAt']
            user = item['snippet']['topLevelComment']['snippet']['authorDisplayName']
            # Extracting comments
            comment = item['snippet']['topLevelComment']['snippet']['textDisplay']
            likeCount = item['snippet']['topLevelComment']['snippet']['likeCount']
            replies.append([published, user, comment, likeCount])
            # counting number of reply of comment
            replycount = item['snippet']['totalReplyCount']
            # if reply is there
            if replycount>0:
                # iterate through all reply
                for reply in item['replies']['comments']:
                    # Extract reply
                    published = reply['snippet']['publishedAt']
                    user = reply['snippet']['authorDisplayName']
                    repl = reply['snippet']['textDisplay']
                    likeCount = reply['snippet']['likeCount']
                    # Store reply is list
                    #replies.append(reply)
                    replies.append([published, user, repl, likeCount])

        # Again repeat
        if 'nextPageToken' in video_response:
            video_response = youtube.commentThreads().list(
                part = 'snippet,replies',
                pageToken = video_response['nextPageToken'],
                videoId = video_id
            ).execute()
        else:
            break
    #endwhile
    return replies
  # Isikan dengan api key
api_key = '-------------------------------'
# Enter video id
# Url video youtube = https://www.youtube.com/watch?v=NhW6OWZjHVU
video_id = "NhW6OWZjHVU" #isikan dengan kode / ID video
# Call function
comments = video_comments(video_id)
#Menyimpan dalam format CSV
df.to_csv('youtube-comments.csv', index=False)
```

Gambar 4.5 Kode Pemrograman *Scraping* Data

```
# isikan dengan api key Anda
api_key = 'AIzaSyC                                                    U'

# Enter video id
# Url video youtube = https://www.youtube.com/watch?v=Nhw6OWZjHVU
video_id = "Nhw6OWZjHVU" #isikan dengan kode / ID video

# Call function
comments = video_comments(video_id)

comments
```

Gambar 4.6 Kode Pemrograman Proses *Scraping*

Gambar 4.7 Hasil Data *Scraping* Komentar

c. Mengubah hasil crawling ke Dataframe. Ini dilakukan untuk merapihkan dataset yang sebelumnya sudah diambil. Dataset atau komentar-komentar tersebut dirapihkan menjadi data table dan diurutkan berdasarkan komentar terbaru.

```
df = pd.DataFrame(comments, columns=['publishedAt', 'authorDisplayName', 'textDisplay', 'likeCount'])
df
```

Gambar 4.8 Kode Pemrograman Pembuatan Table

	publishedAt	authorDisplayName	textDisplay	likeCount
0	2023-02-06T02:42:50Z	Karman wong nanggala	Persiapan boleh. Tp berita berlebihan itu tdk...	0
1	2023-02-05T00:13:18Z	Gery Pasumain	Negara ini masih ada manusia to kenapa takut, ...	0
2	2023-02-04T13:43:18Z	WakJep 40	Salam muhibbah dari Malaysia bang.. Terima Kas...	1
3	2023-02-04T09:47:46Z	Ade Ade	Terima kasih ilmu nya..	0
4	2023-02-04T08:31:56Z	Deni Gayo	Pecogah wen ogoh ini	0
...	...	...	...	...
11519	2022-10-03T06:02:11Z	Muhammad Bakal harkat	ini gw agak takut sih gimna nanti 2023, kaya C...	1
11520	2022-10-03T06:01:00Z	Syuaib Palamani	Terimakasih bang, atas informasi yang sangat b...	4
11521	2022-10-03T06:00:47Z	Om Jep Offcl	Good info	1
11522	2022-10-03T06:00:15Z	Smk Muhamadiyah Bandongan	Kok bisa resesi sih bang	1
11523	2022-10-03T08:05:59Z	akunn buyer	Kan udh dijelaskan	0

11524 rows × 4 columns

Gambar 4.9 Hasil Table *Scraping* Komentar YouTube

d. Terakhir menyimpan hasil crawling ke file CSV. Setelah dataset sudah di rapihkan maka data tersebut akan disimpan dengan bentuk file CSV yang kemudian file tersebut akan diunduh untuk proses analisis sentiment berikutnya.

```
df.to_csv('youtube-comments.csv', index=False)
```

Gambar 4.10 Kode Pemrograman untuk *Convert to* Csv

4.2 *Explore*

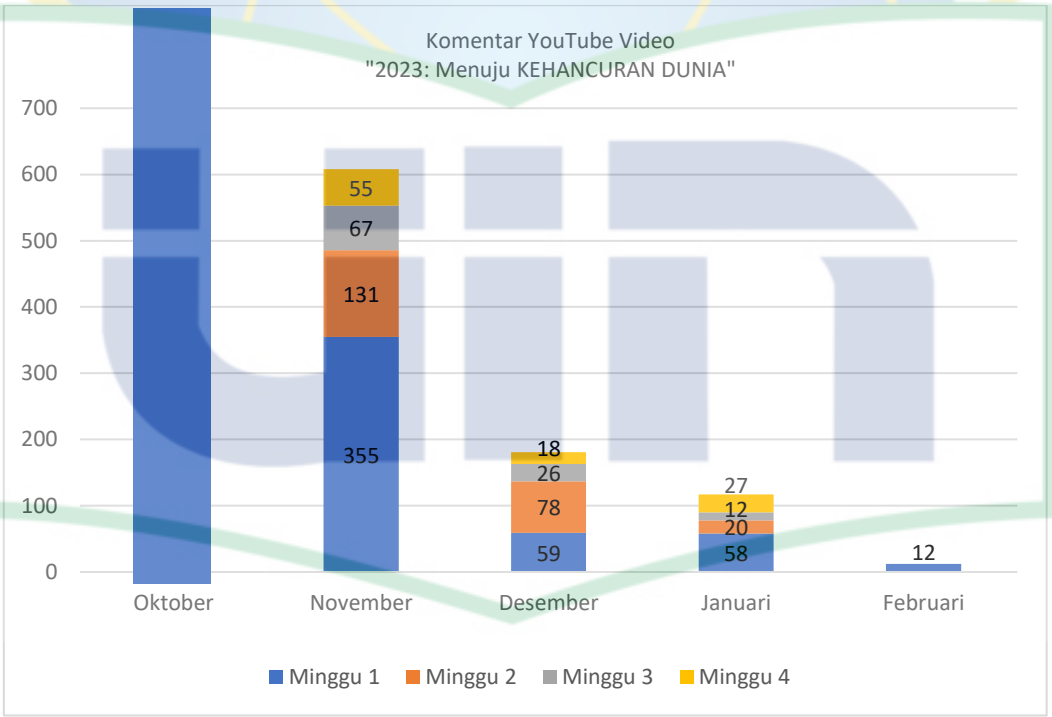
Hasil dari proses pemrograman *Scraping* akan berisikan 4 kolom rincian dari masing – masing komentar YouTube seperti *publishedAt* yaitu waktu dimana pengguna melontarkan komentarnya pada video YouTube ini, *authorDisplayedName* yaitu nama akun yang berkomentar pada video YouTube ini, *textDisplayed* yaitu isi dari komentar video, dan terakhir adalah *likeCount* yaitu jumlah pengguna yang menyukai komentar video tersebut. Pada penelitian ini dari 4 kolom hasil yang sebelumnya didapat, akan dipangkas menjadi 2 kolom saja yang akan ditampilkan yaitu *publishedAt dan textDisplayed*.

	publishedAt	textDisplayed
2	2023-02-06T02:42:50Z	Persiapan boleh. Tp berita berlebihan itu tdk mungkin. Kita baik2ajah. Saya yakin.. Kalau pun iah. Kmungkinan di tahun 2024/2025 itu juga mungkin. Tp untuk 2023. Tdk mungkin saya
3	2023-02-05T00:13:18Z	Negara ini masih ada manusia to kenapa takut, ngak usah ribet yg penting pangan di perkuat
4	2023-02-04T13:43:18Z	Salam muhibbah dari Malaysia bang.. Terima Kasih atas kontennya, sangat mudah difahami dengan anologi yg simple...mogaÂ² membawa manfaat kepada umat manusia..ðŸ«°ðŸ'ŒðŸ'Œ
5	2023-02-04T09:47:46Z	Terima kasih ilmu nya..
6	2023-02-04T08:31:56Z	Pecogah wen ogoh ini
7	2023-02-02T15:50:25Z	Yg bikin jatuh itu si jokowi,...
8	2023-02-02T04:47:17Z	Untuk pembiayaan perang dgn Rusia..As dan uni Eropa..butuh dana besar....nah dr mn akan mrk dapat kl gak dr merampok kekayaan semua negara di dunia...gmn caranya..ya dgn menaikkan
9	2023-02-01T04:51:32Z	alah trik dagangan, bikin konten viral
10	2023-01-31T05:24:16Z	Trus kayak saya hanya penjual sayur keliling gimana nasip kami nanti.
11	2023-01-30T02:15:04Z	Ya bnr aj hancur dunia indo..udh 6 thn di bahas sekarang...orng jakarta koruptor..uang byk..tp bagi uang ke rakyat tdk cukup ..sehari cm 50 rb..Pns Tni Polisi itu gaji buat ksh mkn ..orang tua ..ja
12	2023-01-30T02:09:23Z	Tni sm polisi dr dl gaji buat ngasih mkn orang tuanya ..paroan gaji..ap lg rakyat muskin cm ad lahan sm sawah dia bisa mkn..tp cr uang 50 rb ..cuma buat beli listrik dn gas..mkn cuma nasi sayur
13	2023-01-29T10:05:26Z	Kerak
14	2023-01-28T18:14:01Z	Kasih mahal terus pupuk tu... Biar mati semua sektor pertanian....
15	2023-01-28T02:28:47Z	Dari 2018 hidup dan ekonomi gw udh resesi, jd ya menghindari masalah dengan 1. Jangan punya utang 2. Jangan beli beli ga penting 3. Ngirit 4. Bersyukur 5. Jangan bnyak
16	2023-01-27T05:18:12Z	kerana Sistemnya itu adalah Sistemnya dajal kaya terus bertambah kaya Dan yang miskin terus bertambah miskin itu bertentangan Maka itu yang terjadi dengan Sistemnya tuhan benar adil D
17	2023-01-27T03:02:42Z	Menyeramkan mana ah! Lebay Lo , ini g PP 2023 ??
18	2023-01-25T02:15:05Z	Desa adalah kekuatan ekonomi Selagi desa ga di no1 kan maka tunggulah kehancuran nya
19	2023-01-23T16:36:41Z	Ko , public speakinglu keren terdiastract sama cara pembawaan materinya
20	2023-01-23T15:41:17Z	Akan datang isa almasih yesus kristus
21	2023-01-23T15:40:51Z	Benar hati -hati berdoa bertobat
22	2023-01-23T15:37:30Z	Itu efek dari jus alpukat anak 2 TUHAN banyak yg tertinggal .... Ishak dan ismail di depan akan ada perang dan penyiksaan . Jgn menyangkal isa anak maryam) yesus kristus
23	2023-01-23T15:35:16Z	Benar kedepan nya ada zombie
24	2023-01-23T15:31:31Z	Iya benar. lihat waktu di percepat bye teman teman ku ...

Gambar 4.11 Hasil *Filtering* Komentar

Dataset telah diurutkan dari komentar atau *publishedAt* yang paling terbaru hingga komentar pertama dari video ini. Berdasarkan hasil *scraping* komentar YouTube tersebut didapatkan 11.524 komentar dengan rentang waktu dari tanggal 3 Oktober 2022 hingga 6 Februari 2023. Maka selanjutnya dataset yang sudah didapat akan dibuatkan grafik agar informasi lebih mudah terlihat. Grafik ini akan dibuat berdasarkan waktu *publish*.

Gambar 4.12 Data Komentar YouTube



Gambar 4.13 Data Komentar YouTube (diperbesar)

Berdasarkan grafik atau gambar diatas, terlihat bahwa saat pertama video ini dirilis di *platform* YouTube yaitu pada bulan Oktober 2022, menyita banyak sekali

perhatian pengguna YouTube dengan sekitar 6590 komentar pada minggu pertama bulan oktober, 2361 komentar pada minggu kedua, 997 komentar diminggu ketiga, dan 662 komentar diminggu ke-empat bulan Oktober. Dengan jumlah 10.606 komentar pada bulan Oktober, didapatkan rata rata 942 komentar perhari diminggu pertama video ini keluar sehingga dapat dikatakan video ini sangat lah diperbincangkan di media sosial YouTube umumnya oleh masyarakat Indonesia.

4.3 *Modify*

Tahap ini merupakan tahap disaat dataset dilakukan *text preprocessing* untuk di modifikasi dengan tujuan untuk menjadikan data set terstruktur dan dapat dibaca atau dikenali oleh sistem untuk tahap selanjutnya. Pada tahap ini, peneliti melakukan tahapan-tahapan *text preprocessing case folding, cleaning, tokenize, normalize, stemming, dan stopword removal* pada *tools* rapidminer versi 10.0.0 dan jupyter *notebook.*

4.3.1 *Case Folding*

Lahkah pertama pada tahap *modify* pada penelitian ini adalah *case folding*. Tahap *case folding* dilakukan untuk merubah semua karakter kata dalam data set menjadi huruf kecil. Pada tahap ini dilakukan *case folding* menggunakan python. Dari dataset yang sudah didapatkan sebelumnya, maka dataset tersebut

```
# case folding

df['case_folding'] = df['textDisplay'].str.lower()
df
```

Gambar 4.14 Kode *Case Folding* Python

Setelah dilakukan proses tersebut maka didapatkan hasil data set yang sudah tidak mempunyai lagi huruf kapital.

Tabel 4.1 Contoh Hasil *Case Folding*

Sebelum *Case Folding*	**Sesudah *Case Folding***
Abis Nonton ini gw Sadar, Knpa pandemi 2thn terakhir ini, Paypal ngasi $5 setiap bulan nya unt setiap akun secara cuma2. Biar dolar terus berputar	abis nonton ini gw sadar, knpa pandemi 2thn terakhir ini, paypal ngasi $5 setiap bulan nya unt setiap akun secara cuma2. biar dolar terus berputar

	textDisplay	publishedAt	case_folding
0	Kok bisa resesi sih bang	2022-10-03 06:00:15+00:00	kok bisa resesi sih bang
1	Good info	2022-10-03 06:00:47+00:00	good info
2	Terimakasih bang, atas informasi yang sangat b...	2022-10-03 06:01:00+00:00	terimakasih bang, atas informasi yang sangat b...
3	ini gw agak takut sih gimna nanti 2023, kaya C...	2022-10-03 06:02:11+00:00	ini gw agak takut sih gimna nanti 2023, kaya c...
4	Makasih bang Informasi nya, Sekarang saya mau ...	2022-10-03 06:03:15+00:00	makasih bang informasi nya, sekarang saya mau ...
...	...	...	...
11519	Pecogah wen ogoh ini	2023-02-04 08:31:56+00:00	pecogah wen ogoh ini
11520	Terima kasih ilmu nya..	2023-02-04 09:47:46+00:00	terima kasih ilmu nya..
11521	Salam muhibbah dari Malaysia bang.. Terima Kas...	2023-02-04 13:43:18+00:00	salam muhibbah dari malaysia bang.. terima kas...
11522	Negara ini masih ada manusia to kenapa takut, ...	2023-02-05 00:13:18+00:00	negara ini masih ada manusia to kenapa takut, ...
11523	Persiapan boleh. Tp berita berlebihan itu tdk...	2023-02-06 02:42:50+00:00	persiapan boleh. tp berita berlebihan itu tdk...

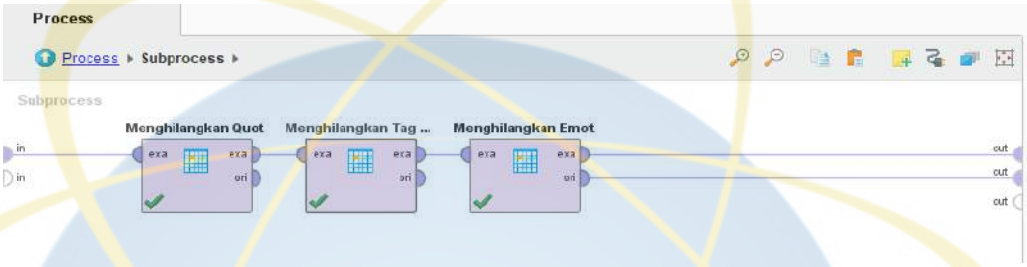
11524 rows × 3 columns

Gambar 4.15 Hasil *Case Folding Dataset*

4.3.2 *Cleaning*

Langkah kedua pada tahap *modify* pada penelitian ini adalah *cleaning*. Tahap *cleaning* didalam penelitian ini terbagi lagi menjadi 2 tahapan yaitu *cleaning* menggunakan RapidMiner dan *cleaning* dengan python. Tahap *cleaning* menggunakan RapidMiner untuk menghapus atribut yang tidak dapat

terbaca oleh python didalam file csv seperti *emoticon* dan tag html yang tercampur dengan komentar didalam suatu teks. Kemudian selanjutnya *cleaning* menggunakan python untuk menghapus atribut data yang tidak memiliki makna dan tidak dibutuhkan seperti url, tanda baca, *mention*, *whitespaces*, dan lain sebagainya.



Gambar 4.16 Alur Proses *Cleaning*

```
# cleaning
import string
import re

def cleaning(komentar):

    #remove ascii
    komentar = komentar.encode('ascii', 'replace').decode('ascii')
    #remove angka
    komentar = re.sub('[0-9]+', '', komentar)
    # remove mention, link, hashtag
    komentar = re.sub('<br.*?>(.+?)</br>',' ',komentar)
    komentar = ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)"," ", komentar).split())
    komentar = re.sub('@[^\s]+','',komentar)
    #remove url
    komentar = re.sub(r'\w+:\/{2}[\d\w-]+(\.[\d\w-]+)*(?:(?:\/[^\s/]*))*', '', komentar)
    #remove tanda baca
    komentar = re.sub(r'[^\w\d\s]+', '', komentar)
    #remove whitespace
    komentar = re.sub('\s+',' ',komentar)


    return komentar
df['cleaning'] = df['case_folding'].apply(cleaning)

# menghapus tweet duplikat
df.drop_duplicates(subset ="cleaning", keep = 'first', inplace = True)
df
```

Gambar 4.17 Proses *Cleaning* dengan Python

Tabel 4.2 Hasil *Cleaning*

Sebelum *Cleaning*	**Sesudah *Cleaning***
“2023 bakal ngeri banget dll.” Sama aja dengan menebarkan fear sehingga orang pada nahan duit ga	2023 bakal ngeri banget dll. Sama aja dengan menebarkan fear

ada yang spend. Resesi ya dimulai dari ketakutan itu sendiri.	sehingga orang pada nahan duit ga ada yang spend. Resesi ya dimulai dari ketakutan itu sendiri.

4.3.3 *Tokenize*

Tahap selanjutnya adalah *tokenize* atau tokenisasi dimana suatu kalimat dari komentar YouTube atau Dataset akan dipecah menjadi potongan kata atau token. Hal ini dilakukan untuk mengetahui kemunculan dari kata tersebut.

Tabel 4.3 Contoh Hasil Tokenisasi

Sebelum Tokenisasi	Sesudah Tokenisasi
kasi tips terutama untuk masyarakat yg menengah kebawah	[‘kasi’, ‘tips’, ‘terutama’, ‘untuk’, ‘masyarakat’, ‘yg’, ‘menengah’, ‘kebawah’]

```
# tokenize
import nltk
nltk.download('punkt')

from nltk.tokenize import word_tokenize

def word_tokenize_wrapper(cleaning):
    return word_tokenize(cleaning)

df['tokenize'] = df['cleaning'].apply(word_tokenize_wrapper)
df
```

Gambar 4.18 Kode Python Tokenisasi

4.3.4 Normalisasi

Tahap berikutnya adalah normalisasi atau *normalize*. Didalam penelitian ini, tahap normalisasi dilakukan untuk menstandarisasi kata yang

memiliki makna yang sama dengan melakukan perubahan penulisan kata yang disingkat dan atau tidak baku. Penelitian ini menggunakan sebuah kamus normalisasi yang didapatkan dari kamus NLP (*Neuro Linguistic Programming*) bahasa Indonesia Resource (Owen,2020).

Table 4.4 Contoh Hasil Normalisasi

Sebelum Normalisasi	Sesudah Normalisasi
[‘kasi’, ‘tips’, ‘terutama’, ‘untuk’, ‘masyarakat’, ‘yg’, ‘menengah’, ‘kebawah’]	[‘kasih’, ‘tips’, ‘terutama’, ‘untuk’, ‘masyarakat’, ‘yang’, ‘menengah’, ‘kebawah’]

```
# normalisasi
import nltk

normalizad_word = pd.read_excel(("normalisasi.xlsx"), engine='openpyxl',)
normalizad_word_dict = {}
for index, row in normalizad_word.iterrows():
    if row[0] not in normalizad_word_dict:
        normalizad_word_dict[row[0]] = row[1]

def normalized_term(document):
    return [normalizad_word_dict[term] if term in normalizad_word_dict else term for term in document]

df['normalize'] = df['tokenize'].apply(normalized_term)
df
```

Gambar 4.19 Kode Normalisasi Python

	textDisplay	publishedAt	case_folding	cleaning	tokenize	normalize
0	Persiapan boleh Tp berita berlebihan itu tdk...	2023-02-06	persiapan boleh tp berita berlebihan itu tdk...	persiapan boleh tp berita berlebihan itu tdk m...	[persiapan, boleh, tp, berita, berlebihan, itu...	[persiapan, boleh, tapi, berita, berlebihan, i...
1	Negara ini masih ada manusia to kenapa takut ...	2023-02-05	negara ini masih ada manusia to kenapa takut ...	negara ini masih ada manusia to kenapa takut n...	[negara, ini, masih, ada, manusia, to, kenapa,...	[negara, ini, masih, ada, manusia, to, kenapa,...
2	Salam muhibbah dari Malaysia bang Terima Kas...	2023-02-04	salam muhibbah dari malaysia bang terima kas...	salam muhibbah dari malaysia bang terima kasih...	[salam, muhibbah, dari, malaysia, bang, terima...	[salam, muhibbah, dari, malaysia, bang, terima...
3	Terima kasih ilmu nya	2023-02-04	terima kasih ilmu nya	terima kasih ilmu nya	[terima, kasih, ilmu, nya]	[terima, kasih, ilmu, nya]
4	Pecogah wen ogoh ini	2023-02-04	pecogah wen ogoh ini	pecogah wen ogoh ini	[pecogah, wen, ogoh, ini]	[pecogah, wen, ogoh, ini]
...	...	...	...	...	...	...
9919	Klo bner tahun gelap Berarti itu tanda ta...	2022-10-03	klo bner tahun gelap berarti itu tanda ta...	klo bner tahun gelap berarti itu tanda tandany...	[klo, bner, tahun, gelap, berarti, itu, tanda,...	[kalau, bner, tahun, gelap, berarti, itu, tand...
9920	Kalo lu ngga percaya Konspirasi apa yang lu e...	2022-10-03	kalo lu ngga percaya konspirasi apa yang lu e...	kalo lu ngga percaya konspirasi apa yang lu ed...	[kalo, lu, ngga, percaya, konspirasi, apa, yan...	[kalau, kamu, tidak, percaya, konspirasi, apa,...
9921	Judul nya serem optimis saja kalo pikiran ...	2022-10-03	judul nya serem optimis saja kalo pikiran ...	judul nya serem optimis saja kalo pikiran di b...	[judul, nya, serem, optimis, saja, kalo, pikir...	[judul, nya, serem, optimis, saja, kalau, piki...
9922	Brti tahun ini harus nyiapin peluru buat disko...	2022-10-03	brti tahun ini harus nyiapin peluru buat disko...	brti tahun ini harus nyiapin peluru buat disko...	[brti, tahun, ini, harus, nyiapin, peluru, bua...	[brti, tahun, ini, harus, mempersiapkan, pelur...
9923	Tutor bulan harus jdi kaya	2022-10-03	tutor bulan harus jdi kaya	tutor bulan harus jdi kaya	[tutor, bulan, harus, jdi, kaya]	[tutor, bulan, harus, jadi, kaya]

Gambar 4.20 Hasil Normalisasi

4.3.5 *Stopword Removal*

Stopword Removal bertujuan untuk menghapus kata-kata umum yang banyak digunakan namun tidak memberikan pengaruh sentimen pada suatu kalimat. Proses *stopword* yang digunakan pada penelitian ini adalah dengan memanfatkan *library* dari Sastrawi yang di dalamnya terdapat *corpus stopword* bahasa Indonesia.

Tabel **4.5 Contoh Hasil *Stopword Removal***

Sebelum *Stopword Removal*	Sesudah *Stopword Removal*
[‘terima’, ‘kasih’, ‘ilmu’, ‘nya’]	[‘terima’, ‘kasih’, ‘ilmu’]

```
# stopword removal
from nltk.corpus import stopwords

list_stopwords = stopwords.words('indonesian')
#tambahkan stopword manual
list_stopwords.extend(['tu', 'uf', 'deh','nak','amp','b','a', 'je', 'x',
                       'sih', 'dos', 'm', 'eh', 'tuh', 'hm', 'nya',
                      'ufufuf', 'lho', 'rm', 'nya', 'ufcc', 'ppv', 'via', 'pon',
                      'dok', 'je', 'gb', 'pa', 'e', ])
sw = set(list_stopwords) - set (['jangan', 'jangankan', 'janganlah', 'tidak', 'tidakkah', 'tidaklah'])


def stopwords_removal(words):
    return [word for word in words if word not in sw]
df['stopword_removal'] = df['normalize'].apply(stopwords_removal)
df
```

Gambar 4.21 Alur Sub Proses *Stopword Removal*

	textDisplay	publishedAt	case_folding	cleaning	tokenize	normalize	stopword_removal
0	Persiapan boleh Tp berita berlebihan itu tdk...	2023-02-06	persiapan boleh tp berita berlebihan itu tdk...	persiapan boleh tp berita berlebihan itu tdk m...	[persiapan, boleh, tp, berita, berlebihan, itu...	[persiapan, boleh, tapi, berita, berlebihan, i...	[persiapan, berita, tidak, ajah, iah, kmungkin...
1	Negara ini masih ada manusia to kenapa takut ...	2023-02-05	negara ini masih ada manusia to kenapa takut ...	negara ini masih ada manusia to kenapa takut n...	[negara, ini, masih, ada, manusia, to, kenapa,...	[negara, ini, masih, ada, manusia, to, kenapa,...	[negara, manusia, to, takut, ngak, ribet, pang...
2	Salam muhibbah dari Malaysia bang Terima Kas...	2023-02-04	salam muhibbah dari malaysia bang terima kas...	salam muhibbah dari malaysia bang terima kasih...	[salam, muhibbah, dari, malaysia, bang, terima...	[salam, muhibbah, dari, malaysia, bang, terima...	[salam, muhibbah, malaysia, bang, terima, kasi...
3	Terima kasih ilmu nya	2023-02-04	terima kasih ilmu nya	terima kasih ilmu nya	[terima, kasih, ilmu, nya]	[terima, kasih, ilmu, nya]	[terima, kasih, ilmu]
4	Pecogah wen ogoh ini	2023-02-04	pecogah wen ogoh ini	pecogah wen ogoh ini	[pecogah, wen, ogoh, ini]	[pecogah, wen, ogoh, ini]	[pecogah, wen, ogoh]
...	...	...	...	...	...	...	...
9919	Klo bner tahun gelap Berarti itu tanda ta...	2022-10-03	klo bner tahun gelap berarti itu tanda ta...	klo bner tahun gelap berarti itu tanda tandany...	[klo, bner, tahun, gelap, berarti, itu, tanda,...	[kalau, bner, tahun, gelap, berarti, itu, tand...	[bner, gelap, tanda, tandanya, minggu, minggu,...
9920	Kalo lu ngga percaya Konspirasi apa yang lu e...	2022-10-03	kalo lu ngga percaya konspirasi apa yang lu e...	kalo lu ngga percaya konspirasi apa yang lu ed...	[kalo, lu, ngga, percaya, konspirasi, apa, yan...	[kalau, kamu, tidak, percaya, konspirasi, apa,...	[tidak, percaya, konspirasi, edukasi, in, view...

Gambar 4.22 Hasil *Stopword Removal*

4.3.6 *Stemming*

Tahap terakhir didalam *modify* pada penelitian ini adalah tahap *stemming*. Tahap *stemming* dilakukan oleh peneliti dengan tujuan untuk mendapatkan kata dasar dari suatu kata menggunakan *library* Sastrawi. Dalam kata lain kata pada dataset dari tahap sebelumnya diubah ke kata dasar sesuai KBBI.

```
# stemming
import Sastrawi
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

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

def stemmed_wrapper(term):
    return stemmer.stem(term)
df['stemming'] = df['stopword_removal'].apply(lambda x: [stemmer.stem(y) for y in x])
df
```

Gambar 4.23 Alur Sub Proses *Stemming*

4.4 Model

Tahap Pemodelan atau tahap model didalam penelitian ini melakukan pelebelan dengan 3 pemodelan yaitu dengan metode *lexicon based* untuk mendapatkan label kelas didalam dataset dan menggunakan 2 pemodelan klasifikasi dengan metode Naïve Bayes dan K-Nearest Neighbor.

4.4.1 *Lexicon Based*

Untuk melakukan klasifikasi didalam penelitian ini digunakanlah metode *lexicon based* dilakukan pada kelas dataset. Terdapat 2 kelas yaitu kelas positif dan kelas negatif didalam suatu komentar atau dokumen pada dataset. Kata kata pada dataset akan dibandingkan dengan dokumen kamus *lexicon*. Perbandingan kata tersebut akan diberikan nilai atau *score* jika kata pada komentar atau dokumen didalam dataset terdapat kesamaan dengan kata yang ada didalam kamus *lexicon*. Jumlah nilai atau *score* ini akan menentukan komentar tersebut termasuk didalam label positif atau negatif. Kamus *lexicon* yang digunakan bersumber dari Koto (2017) yaitu *Indonesian Sentiment* (Inset). Terdapat kurang lebih 10.250 kata yang diberi nilai dari -5 hingga +5.

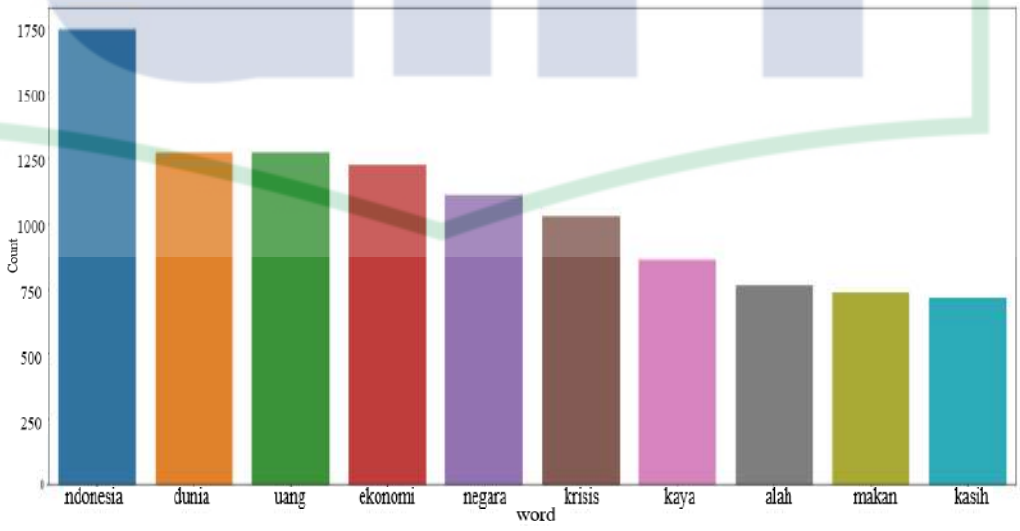
Kemudian peneliti juga akan menambahkan beberapa kata pada Inset lexicon yang bersumber dari github (Martua, 2020).

	A	B	C
1	word	weight	number_of_words
2	hai	3	1
3	merekam	2	1
4	ekstensif	3	1
5	paripurna	1	1
6	detail	2	1
7	pernik	3	1
8	belas	2	1
9	welas	4	1
10	kabung	1	1
11	rahayu	4	1
12	maaf	2	1
13	hello	2	1
14	promo	3	1
15	terimakasih	5	1
16	cover	3	1
17	mohon	2	1
18	mengawal	2	1
19	statistik	1	1
20	keluangan	3	1
21	jalan terbuka	3	2
22	banyaknya	3	1
23	lebar	3	1
24	bentang	1	1
25	hendaknya	1	1

	A	B	C
10186	anjay	-5	1
10187	asu	-5	1
10188	asoe	-5	1
10189	babi	-5	1
10190	biadap	-5	1
10191	biji	-5	1
10192	bajingan	-5	1
10193	banci	-5	1
10194	bangsat	-5	1
10195	bego	-5	1
10196	bengak	-5	1
10197	berak	-5	1
10198	bokong	-5	1
10199	bodoh	-5	1
10200	bongak	-5	1
10201	edan	-5	1
10202	fak	-5	1
10203	fuck	-5	1
10204	fakboi	-5	1
10205	fap	-5	1
10206	gigolo	-5	1
10207	goblok	-5	1
10208	gila	-5	1
10209	gilo	-5	1
10210	homo	-5	1

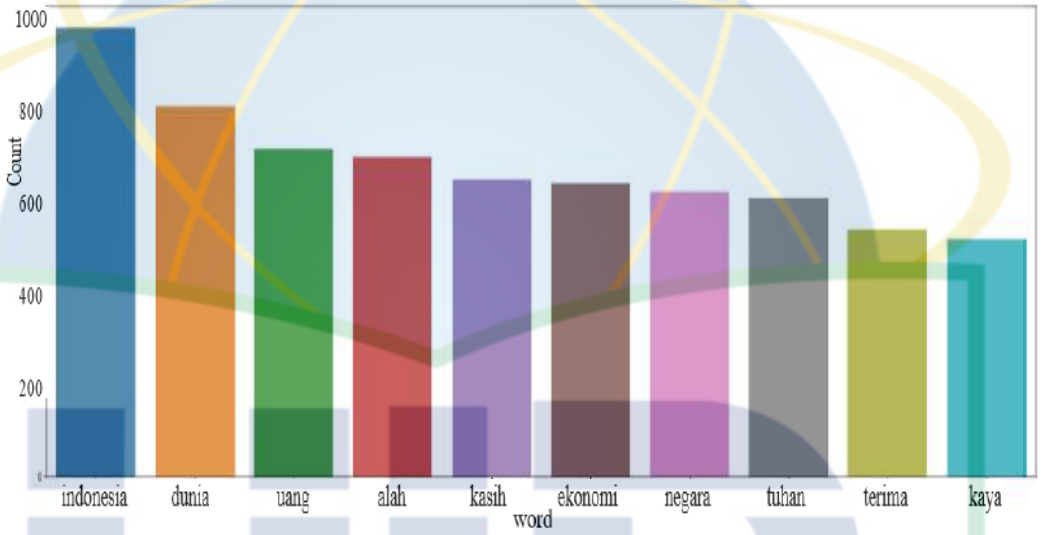
Gambar 4.24 Kamus *Lexicon*

Selanjutnya dari hasil pemberian label dengan menggunakan kamus *lexicon based*, terdapat 8.075 komentar positif dan juga terdapat 3.607 komentar negatif. Dari hasil tersebut menghasilkan kata-kata yang sering diucapkan didalam komentar yang tervisualisasikan dari *word frequency*



Gambar 4.25 Hasil *WordFreq*

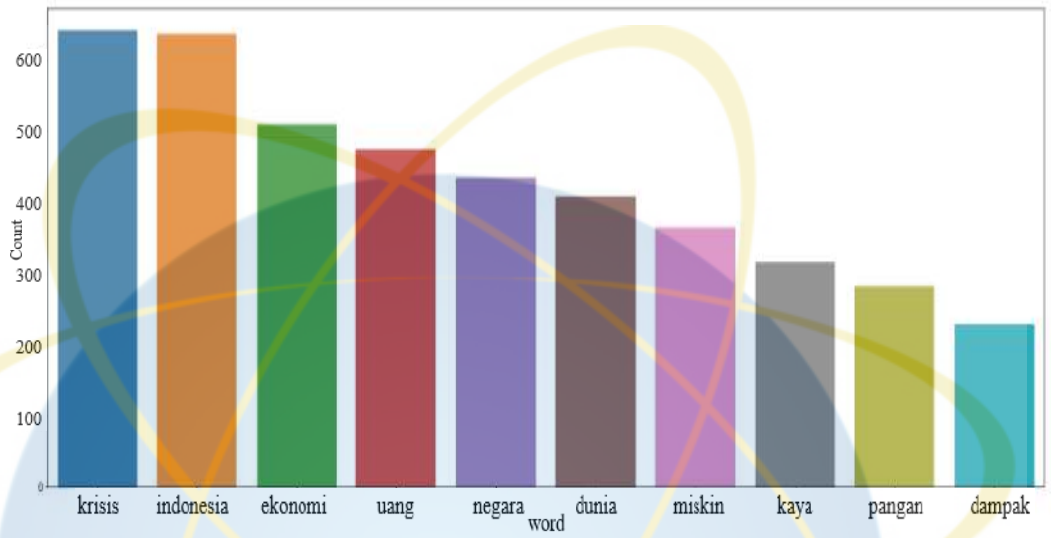
Dari gambar tersebut tervisualisasikan bahwa kata yang sering terucap dalam komentar didalam dataset adalah kata Indonesia kemudian dunia, uang, ekonomi dan sebagainya. Dataset yang sudah diklasifikasikan ke dalam positif dan negatif selanjutnya juga dapat divisualisasikan dalam bentuk grafik dan *wordcloud*. Frekuensi kemunculan kata dari kelas positif yang sering muncul setidaknya 10 kata pertama adalah Indonesia, dunia, uang, alah, kasih, ekonomi, negara, tuhan, terima, kaya.



Gambar 4.26 Hasil *Word Frequency* Positif

Gambar 4.27 Hasil *Word Cloud* Positif

Sedangkan frekuensi kemunculan kata dari kelas negatif yang sering muncul setidaknya 10 kata pertama adalah Indonesia, krisis, ekonomi, uang, negara, miskin, dunia, kaya, pangan, dan makan.



Gambar 4.28 Hasil *Word Frequency* Negatif

Gambar 4.29 Hasil *Word Cloud* Negatif

4.4.2 Naïve Bayes

Pada tahap ini peneliti akan membagi dataset menjadi data latih (*training*) dan data uji (*test*). Peneliti akan membagi beberapa rasio perbandingan percobaan data latih dan data uji ini yaitu dari 70% data latih 30% data uji, kemudian 80% data latih dan 20% data uji, dan yang terakhir 90% data latih dan 10% data uji berdasarkan penelitian dari Gormantara pada tahun 2020 (Gormantara 2020).

Tabel 4.6 Perbandingan Data Latih dan Data Uji Naïve Bayes

Data Latih	Data Uji
70%	30%
80%	20%
90%	10%

4.4.3 K-Nearest Neighbor

Pada tahap ini peneliti juga akan menggunakan konsep yang sama dengan algoritma sebelumnya yaitu Naïve Bayes. Peneliti akan membagi 2 data menjadi data latih dan data uji. Berdasar dengan penelitian sebelumnya yaitu Gormantara pada tahun 2020, peneliti akan membagi rasio data latih dan data uji dari 70% data latih 30% data uji, 80% data latih 20% data uji, dan 90% data latih dan 10% data uji (Gormantara 2020).

Tabel 4.7 Perbandingan Data Latih dan Data Uji K-Nearest Neighbor

Data Latih	Data Uji
70%	30%
80%	20%
90%	10%

Kemudian untuk algoritma K-Nearest Neighbor pada penelitian ini dilakukan uji variasi nilai k. Nilai k akan dicari yang terbaik dari bantuan algoritma KNN itu sendiri, pombobotan TF-IDF dan 10-*fold cross validation* untuk mendapatkannya. Nilai k terbaik akan dilihat dari nilai *error* terkecil. Variasi k yang akan diuji yaitu 10 nilai ganjil pertama dimulai dari k=1 sampai k=19.

4.5 Asses

Tahap terakhir yaitu tahap Asses merupakan tahap untuk dilakukannya evaluasi dari setiap model pada penelitian ini. Hasil evaluasi pada penelitian ini berisikan nilai-nilai *confusion matrix*, nilai akurasi, presisi, recall dan f1-score. Metode K-Nearest Neighbor menggunakan 10-*fold cross validation* dalam pencarian nilai variasi k terbaik yang selanjutnya akan digunakan didalam model.

4.5.1 Naïve Bayes

Pada metode Naïve Bayes didalam penelitian ini, seperti yang sudah di utarakan sebelumnya didalam tahap model, penelitian ini membagi 3 untuk data latih dan data uji dan kemudian diolah dengan metode Naïve Bayes. Berikut merupakan hasil dari pembagian dataset data latih:data uji.

Tabel 4.8 Hasil Akurasi, Presisi, *Recall*, *f1-score* NBC

Data Latih : Data uji	Akurasi	Presisi	*Recall*	*f1-score*
70:30	69.43%	69.73%	68.8%	68.78%
80:20	68.18%	68.67%	67.65%	67.51%
90:10	68.23%	68.85%	67.62%	67.45%

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import TimeSeriesSplit, GridSearchCV

def standardize(X_train_vectors, X_test_vectors):
    '''Function used to column standardize any given matrix'''
    from sklearn.preprocessing import StandardScaler
    scalar = StandardScaler(with_mean=False)
    scalar.fit(X_train_vectors)
    X_train_vectors = scalar.transform(X_train_vectors)
    X_test_vectors = scalar.transform(X_test_vectors)
    print("The shape of the X_train_vectors is : {}".format(X_train_vectors.shape))
    print("The shape of the X_test_vectors is : {}".format(X_test_vectors.shape))
    return (X_train_vectors, X_test_vectors)

def performance(nb_classifier, vectorizationType, X_train, y_train, X_test, y_test, optimal_alpha, mse): #MSE : Mean Squared Loss
    '''Function to measure the various performance metrics for a given model.'''
    print("\n'''PERFORMANCE EVALUATION'''")
    print("\n\nDetailed report for the {} Vectorization.".format(vectorizationType))

    #Predict the labels for the test set.
    y_pred = nb_classifier.predict(X_test)

    #Evaluate the accuracy of the model on test set
    test_accuracy = accuracy_score(y_test, y_pred, normalize=True) * 100
    points = accuracy_score(y_test, y_pred, normalize=False)
    print('\nThe number of accurate predictions out of {} data points on unseen data is {}'.format(X_test.shape[0], points))
    print('Accuracy of the {} model on unseen data is {} %'.format(vectorizationType, np.round(test_accuracy,2)))

    #Get the precision, recall and F1 score for this model.
    print("Precision of the {} model on unseen data is {}".format(vectorizationType, np.round(metrics.precision_score(y_test ,y_
    print("Recall of the {} model on unseen data is {}".format(vectorizationType, np.round(metrics.recall_score(y_test ,y_pred, a
    print("F1 score of the {} model on unseen data is {}".format(vectorizationType, np.round(metrics.f1_score(y_test ,y_pred, ave

    #Classification Report
    print ('\nClasification report for {} model : \n'.format(vectorizationType))
    print(metrics.classification_report(y_test,y_pred))

    #Inference
    print("\nOf all the reviews that the model has predicted to be positive, {}% of them are actually positive.".format(np.round(
    print("Of all the reviews that are actually positive, the model has predicted {}% of them to be positive.".format(np.round(me
```

```
    #Inference
    print("\nOf all the reviews that the model has predicted to be positive, {}% of them are actually positive.".format(np.round(
    print("Of all the reviews that are actually positive, the model has predicted {}% of them to be positive.".format(np.round(me

    #Save the below list for later use to display model information
    info_model_NB = [vectorizationType, optimal_alpha, np.round(np.array(mse).mean(),4), np.round(1-metrics.accuracy_score(y_test
    with open('info_model_NB.txt', 'a') as filehandle:
        filehandle.writelines("%s " % iterator for iterator in info_model_NB)
        filehandle.writelines("\n")

    #Get the confusion matrix for the running model
    print("\nFind below the confusion matrix for {} model.".format(vectorizationType))
    sciplot.plot_confusion_matrix(y_test ,y_pred)

    #Free memory allocations
    del(X_train, y_train, X_test, y_test, vectorizationType, y_pred, nb_classifier)

def get_GridSearchCV_estimator(vectorizationType, X_train, y_train, X_test, y_test):
    '''This function will determine the best hyperparameters using TimeSeriesSplit CV and Grid Search, using 10 fold cross valida
    from sklearn.model_selection import TimeSeriesSplit
    alphas = np.logspace(0, 2, 20)
    tuned_parameters = [{'alpha': alphas}]
    n_folds = 10
    model = MultinomialNB()
    gsearch_cv = GridSearchCV(estimator=model, param_grid=tuned_parameters, cv=10, scoring='f1', n_jobs=6)
    gsearch_cv.fit(X_train, y_train)
    print("\nGridSearchCV completed for {} model!".format(vectorizationType))
    print("Best estimator for {} model : ".format(vectorizationType), gsearch_cv.best_estimator_)
    print("Best Score for {} model : ".format(vectorizationType), gsearch_cv.best_score_)
    return gsearch_cv

def plot_errors(gsearch_cv):
    '''This function is used to plot the curve for mean squared errors vs alpha values'''
    #Get cross validation scores. Here we obtain the alpha values and their corresponding mean test scores.
    cv_result = gsearch_cv.cv_results_
    mts = cv_result["mean_test_score"]       #list that will hold the mean of cross validation accuracy scores for each alpha
    alphas = cv_result["params"]

    alpha_values = []                        #list that will hold all the alpha values that the grid search cross validator tri
    for i in range(0,len(alphas)):
        alpha_values.append(alphas[i]["alpha"])
```

```
    #Changing accuracy to mean squared error. **error = 1 - accuracy ; error = Cross Validation Errors, accuracy = Cross Validat
    mse = [1 - x for x in mts]

    #Determining best alpha from errors. 'alpha' will be best for the lowest value for error
    optimal_alpha = alpha_values[mse.index(min(mse))] #Laplace smoothing
    print('The optimal value of alpha is : {}'.format(optimal_alpha))

    #Plot error vs alpha values
    plt.figure(figsize=(35,8))
    plt.plot(alpha_values , mse, color='green', linestyle='dashed', linewidth=2, marker='o', markerfacecolor='red', markersize=1
    for xy in zip(alpha_values, np.round(mse,3)):
        plt.annotate('(%s, %s)' % xy, xy=xy, textcoords='data')
    plt.title('Plot for Errors vs Alpha Values')
    plt.xlabel('Values of Alpha')
    plt.ylabel('Errors')
    plt.show()

    return (optimal_alpha,mse)

def naive_bayes_algorithm(X_train, y_train, X_test, y_test, vectorizationType, vectorizer_object):
    '''This function splits the dataset into training set and test sets. The test data remains untouched.
    A time series 10 fold cross validation is performed on the train data and the value of optimal alpha is calculated.
    The dataset is then trained with this value of optimal alpha.
    Finally the Naive Bayes model is used to predict its accuracy on the future unseen test set.'''

    #Perform 10-fold cross validation on the train set
    print("Starting Cross Validation steps...")
    gsearch_cv = get_GridSearchCV_estimator(vectorizationType, X_train, y_train, X_test, y_test)

    #Plot the graphical representation of the mean squared error vs the alpha values obtained during cross validation.
    optimal_alpha, mse = plot_errors(gsearch_cv)

    #Initialize the Naive Bayes constructor using alpha = optimal_alpha
    nb_classifier = gsearch_cv.best_estimator_

    #Fit the model to the train set using optimal alpha
    nb_classifier.fit(X_train, y_train)

    #Evaluate the model's performance
    performance(nb_classifier, vectorizationType, X_train, y_train, X_test, y_test, optimal_alpha, mse)
```

Gambar 4.30 Kode Python *Naïve Bayes*

Kemudian untuk hasil dari *confusion matrix* dengan metode Naïve Bayes pada rasio 90%:10% didapat bahwa prediksi benar sentimen positif (*true positive*) ada sebanyak 1416 data dan prediksi benar untuk sentimen negatif (*true negative*) sebanyak 930 data.

Tabel 4.9 Tabel *Confusion Matrix Naïve Bayes*

Hasil Aktual	Nilai Prediksi	
	Negatif	Positif
Negatif	930	654
Positif	379	1416

Mengacu pada persamaan (2.1), (2.2), (2.3) dan perhitungan akurasi, presisi, *recall* dan *f1-score*, maka nilai dari kinerja metode Naïve Bayes sebagai berikut :

$$\text{Akurasi} = \frac{\mathbf{TP + TN}}{\mathbf{TP + TN + FP + FN}} \text{ X } 100\% = \frac{\mathbf{1416 + 930}}{\mathbf{1416 + 930 + 654 + 379}} \text{ X } 100\%$$

$$= 69.42\%$$

$$\text{Presisi Positif} = \frac{\mathbf{TP}}{\mathbf{TP + FP}} \text{ X } 100\% = \frac{\mathbf{1416}}{\mathbf{1416 + 654}} \text{ X } 100\% = 68.40\%$$

$$\text{Presisi Negatif} = \frac{\mathbf{TN}}{\mathbf{TN + FN}} \text{ X } 100\% = \frac{\mathbf{930}}{\mathbf{930 + 379}} \text{ X } 100\% = 71.04\%$$

$$\text{Recall Positif} = \frac{\mathbf{TP}}{\mathbf{TP + FN}} \text{ X } 100\% = \frac{\mathbf{1416}}{\mathbf{1416 + 379}} \text{ X } 100\% = 78.89\%$$

$$\text{Recall Negatif} = \frac{\mathbf{TN}}{\mathbf{TN + FP}} \text{ X } 100\% = \frac{\mathbf{930}}{\mathbf{930 + 654}} \text{ X } 100\% = 58.71\%$$

$$\textit{f1-score} \text{ Positif} = 2 \text{ X } \frac{\mathbf{\text{presisi positif x recall positif}}}{\mathbf{\text{presisi positif + recall positif}}} = 2 \text{ X } \frac{\mathbf{0.6840 \text{ x } 0.7889}}{\mathbf{0.6840 + 0.7889}} = 73.27\%$$

$$\textit{f1-score} \text{ Positif} = 2 \text{ X } \frac{\mathbf{\text{presisi negatif + recall negatif}}}{\mathbf{\text{presisi negatif x recall negatif}}} = 2 \text{ X } \frac{\mathbf{0.7104 \text{ x } 0.5871}}{\mathbf{0.7104 + 0.5871}} = 64.29\%$$

Kemudian setelah mendapat rasio dengan nilai akurasi terbaik, maka selanjutnya dilakukan *cross validation*. *Cross validation* diperlukan untuk mengetahui kinerja minimum dan maksimum yang didapat.

Tabel 4.10 Hasil 10-*folds Cross Validation*

n-fold	Akurasi	Presisi	*Recall*	*f1-score*
1	68%	68%	67%	67%
2	68%	68%	67%	67%
3	68%	68%	66%	66%
4	70%	70%	69%	69%
5	73%	73%	72%	72%
6	70%	70%	69%	69%
7	72%	71%	70%	71%
8	71%	71%	70%	70%
9	69%	69%	68%	68%
10	68%	68%	67%	67%

4.5.2 K-Nearest Neighbor

Metode selanjutnya adalah metode K-Nearest Neighbor (KNN). Pada penelitian ini metode KNN menggunakan beberapa bagian data latih dan data uji seperti yang sebelumnya sudah dijelaskan (tabel 4.7). Untuk menentukan variasi k terbaik, data latih diolah menggunakan algoritma KNN, pembobotan TF-IDF dan 10-*fold validation*. Dataset dijalankan sebanyak 3 kali sesuai dengan skenario yang sebelumnya sudah ditentukan yakni pada tabel 4.7 dengan bahasa pemrograman python variasi k dicari. Hasil uji variasi k dengan pembagian dataset 70%:30% memiliki optimal k=11 dengan plot *error* = 0.415, dataset 80%:20% dengan plot *error* = 0.415, dan dataset 90%:10%

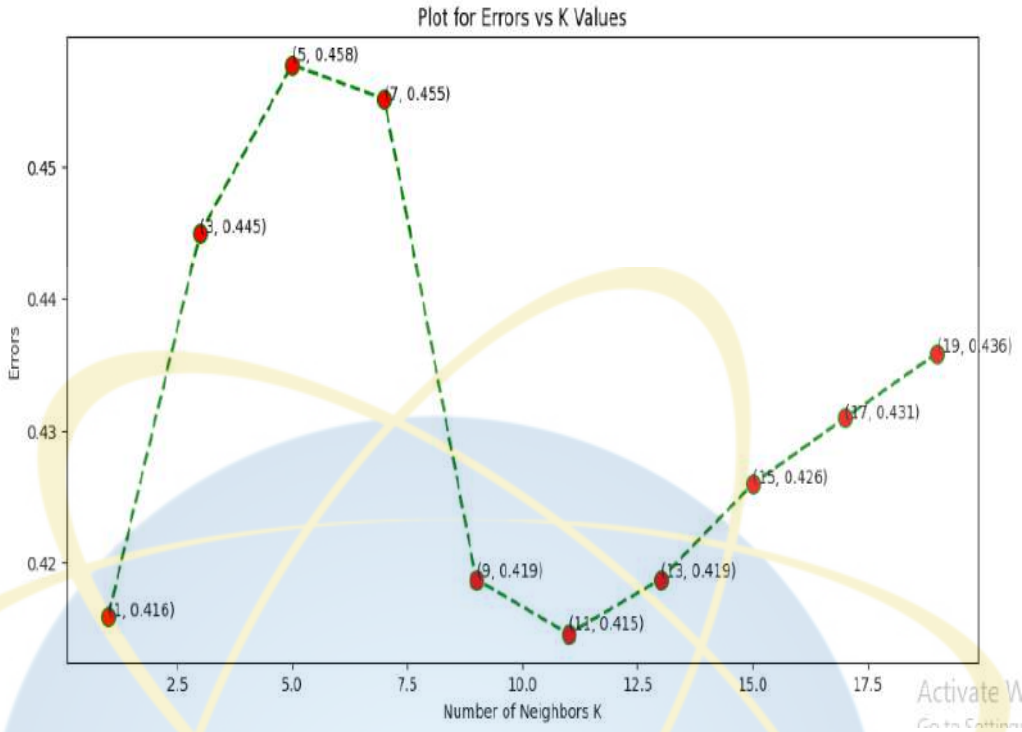
dengan plot *error* = 0.406. Berikut merupakan gambar dari ketiga rasio dengan grafik k terbaik berdasarkan nilai error terkecil.

```
#Classifier Data
def knn_algorithm(X_train, y_train, X_test, y_test, vectorizationType):
    '''This function splits the dataset into training set and test sets. The test data remains untouched.
    A 10 fold cross validation is performed on the train data and the value of optimal K is calculated.
    The dataset is then trained with this value of optimal k.
    Finally the knn model is used to predict its accuracy on the future unseen test set.'''

    X_train = X_train ; y_train = y_train #Train dataframe
    X_test = X_test ; y_test = y_test #Test Dataframe

    #algorithms = ['brute','kd_tree']
    algorithms = ['brute']

    for algo in algorithms:

        print("\nStarting Cross Validation steps for {} model.".format(vectorizationType, algo.upper()))

        #Creating an odd number list of different K values for KNN.
        k_values = list(np.arange(1,20,2))

        #Create an empty list that will hold the mean of cross validation accuracy scores for each value of k in the CV step.
        cross_val_scores = []

        if algo == 'kd_tree':
            svd = TruncatedSVD(n_components = 100)
            X_train = svd.fit_transform(X_train)
            X_test = svd.fit_transform(X_test)

        #Perform 10-fold cross validation on the train set
        for k in k_values:
            knn_classifier = KNeighborsClassifier(n_neighbors=k, weights='distance', algorithm=algo, p=2, metric='minkowski', n_
            accuracies = cross_val_score(knn_classifier, X_train, y_train, cv=10, scoring='accuracy')
            cross_val_scores.append(accuracies.mean())
            #print("Cross validation completed using k = {}".format(k))

        #Changing accuracy to error. **error = 1 - accuracy
        errors = [1 - x for x in cross_val_scores]

        #Determining best k from errors. K will be best for the lowest value for error.
        optimal_k = k_values[errors.index(min(errors))]
        print('\nThe optimal number of neighbors is : {}'.format(optimal_k))
```

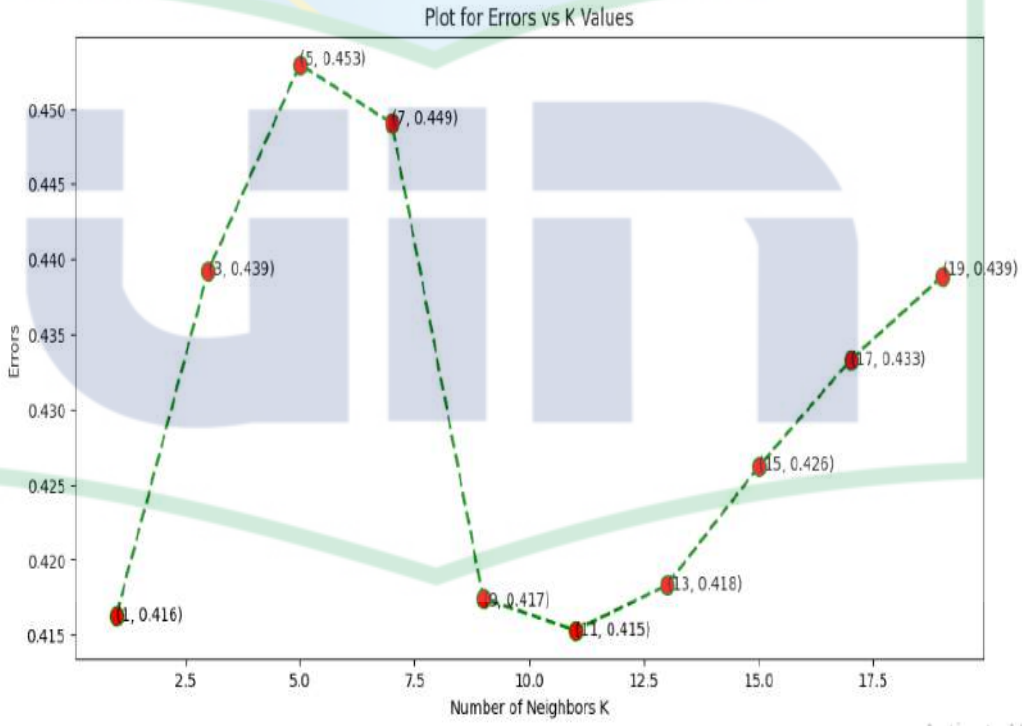
```
#Plot errors vs k values
plt.figure(figsize=(12,6))
plt.plot(k_values , errors, color='green', linestyle='dashed', linewidth=2, marker='o', markerfacecolor='red', markersize
for xy in zip(k_values, np.round(errors,3)):
    plt.annotate('(%s, %s)' % xy, xy=xy, textcoords='data')
plt.title('Plot for Errors vs K Values')
plt.xlabel('Number of Neighbors K'.format(algo.upper()))
plt.ylabel('Errors')
plt.show()

print("The error for each k value: {}".format(np.round(errors,3)))


'''Train the model using the optimal value of k found from the previous step and evaluate it's accuracy on the test set(

#Initialize the KNN model, where k = optimal_k
knn_classifier = KNeighborsClassifier(n_neighbors=optimal_k, weights='distance', algorithm='kd_tree', p=2, metric='minkow

#Fit the model to the train set
knn_classifier.fit(X_train, y_train)

#Predict the labels for the test set.
y_pred = knn_classifier.predict(X_test)

'''PERFORMANCE EVALUATION'''

print("\n'''PERFORMANCE EVALUATION FOR {} model'''".format(vectorizationType))

print("\n\nDetailed report for the {} Vectorization:".format(vectorizationType))

#Evaluate the accuracy of the model on test set
test_accuracy = accuracy_score(y_test, y_pred, normalize=True) * 100
points = accuracy_score(y_test, y_pred, normalize=False)
print('The number of accurate predictions out of {} data points on unseen data for K = {} is {}'.format(X_test_vectors.sh
print('\nAccuracy of the KNN model on unseen data for K = {} is {} %'.format(optimal_k, np.round(test_accuracy,2)))

#Get the precision, recall and F1 score for this model.
print("Precision of the KNN model on unseen data for K = {} is {}".format(optimal_k, np.round(metrics.precision_score(y_t
print("Recall of the KNN model on unseen data for K = {} is {}".format(optimal_k, np.round(metrics.recall_score(y_test,y
print("F1 score of the KNN model on unseen data for K = {} is {}".format(optimal_k, np.round(metrics.f1_score(y_test ,y_p
```

```
'''PERFORMANCE EVALUATION'''

print("\n'''PERFORMANCE EVALUATION FOR {} model'''".format(vectorizationType))

print("\n\nDetailed report for the {} Vectorization:".format(vectorizationType))

#Evaluate the accuracy of the model on test set
test_accuracy = accuracy_score(y_test, y_pred, normalize=True) * 100
points = accuracy_score(y_test, y_pred, normalize=False)
print('The number of accurate predictions out of {} data points on unseen data for K = {} is {}'.format(X_test_vectors.sh
print('\nAccuracy of the KNN model on unseen data for K = {} is {} %'.format(optimal_k, np.round(test_accuracy,2)))

#Get the precision, recall and F1 score for this model.
print("Precision of the KNN model on unseen data for K = {} is {}".format(optimal_k, np.round(metrics.precision_score(y_t
print("Recall of the KNN model on unseen data for K = {} is {}".format(optimal_k, np.round(metrics.recall_score(y_test ,y
print("F1 score of the KNN model on unseen data for K = {} is {}".format(optimal_k, np.round(metrics.f1_score(y_test ,y_p

#Classification Report
print ('\nClasification report for {} model: \n'.format(vectorizationType))
print(metrics.classification_report(y_test,y_pred))

#Inference
print("\nOf all the reviews that the model has predicted to be positive, {}% of them are actually positive.".format(np.ro
print("Of all the reviews that are actually positive, the model has predicted {}% of them to be positive.".format(np.roun

#Get the confusion matrix for the running model
print("\nFind below the confusion matrix for {} model.".format(vectorizationType))
sciplot.plot_confusion_matrix(y_test ,y_pred)

#Save the below list for later use to display model information
info_model_KNN = [vectorizationType, optimal_k, np.round(np.array(errors).mean(),4), np.round(1-metrics.accuracy_score(y_
with open('info_model_KNN.txt', 'a') as filehandle:
    filehandle.writelines("%s " % iterator for iterator in info_model_KNN)
    filehandle.writelines("\n")

#Freeing memory allocations
del(X_train, y_train, X_test, y_test, y_pred, knn_classifier)
```

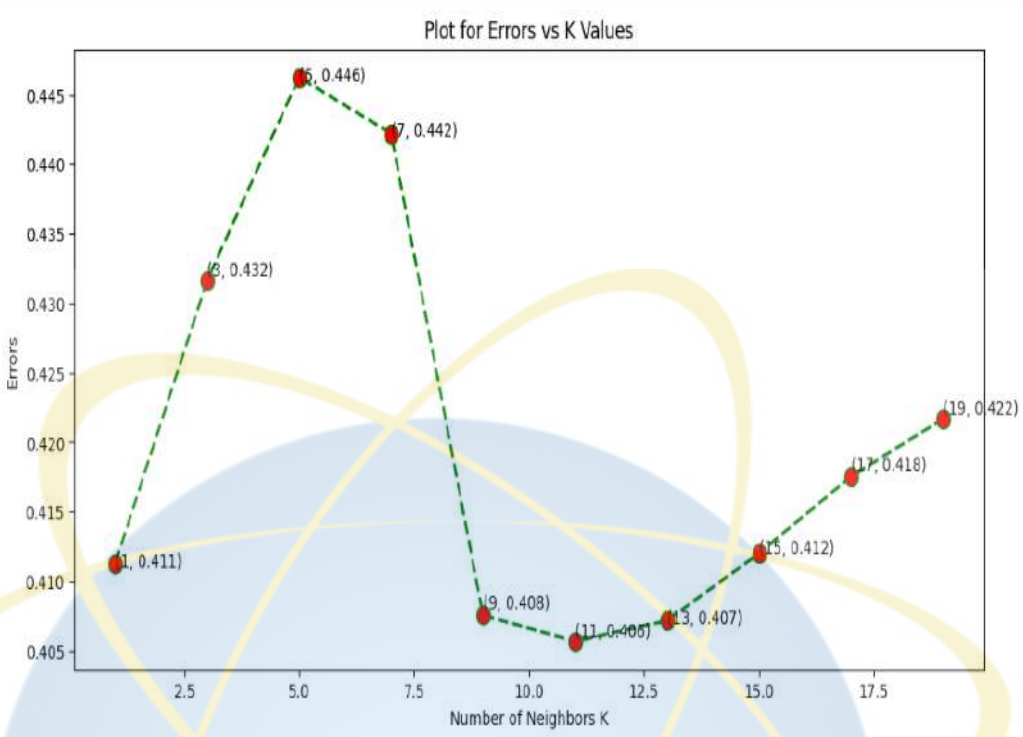
Gambar 4.31 Kode Python Klasifikasi KNN



Gambar 4.32 Hasil Data Uji Rasio 70:30 Variasi k



Gambar 4.33 Hasil Data Uji Rasio 80:20 Variasi k



Gambar 4.34 Hasil Data Uji Rasio 90:10 Variasi k

Nilai k terbaik sudah didapatkan, setelah itu adalah kembali melakukan uji akurasi, presisi, *recall*, dan *f1-score* dengan ketiga rasio data yaitu 70%:30%, 80%:20% dan 90%:10%. Algoritma perhitungan jarak yang digunakan pada metode K-Nearest Neighbor ini adalah *Euclidean Distance*. Berikut merupakan hasil akurasi, presisi, *recall* dan *f1-score* dari masing-masing rasio.

Tabel 4.11 Hasil Akurasi, Presisi, *Recall*, dan *f1-score* KNN

Data Latih : Data uji	**k**	**Akurasi**	**Presisi**	***Recall***	***f1-score***
70:30	11	63.24%	63.9%	63.71%	63.2%
80:20	11	63.06%	73.53%	64.57%	59.79%
90:10	11	65.45%	74.67%	66.94%	63.05%

Kinerja terbaik terdapat pada data dengan rasio 90%:10% dengan k=11. Nilai akurasi rasio tersebut adalah 65.45%. Kemudian untuk hasil dari *confusion matrix* pada rasio 90%:10% ini didapatkan prediksi benar pada sentimen positif (*true positive*) adalah 225 data. Sedangkan prediksi benar untuk sentimen negatif (true negative) adalah 512 data.

Tabel 4.12 Hasil *Confusion Matrix* KNN

Hasil Aktual	Nilai Prediksi	
	Negatif	Positif
Negatif	512	22
Positif	367	225

Mengacu pada persamaan yang sebelumnya dan perhitungan akurasi, presisi, *recall* dan *f1-score*, maka nilai dari kinerja metode K-Nearest Neighbor sebagai berikut :

$$\text{Akurasi} = \frac{\mathbf{TP + TN}}{\mathbf{TP + TN + FP + FN}} \text{ X } 100\% = \frac{\mathbf{512 + 225}}{\mathbf{512 + 225 + 22 + 367}} \text{ X } 100\%$$

$$= 65.45\%$$

$$\text{Presisi Positif} = \frac{\mathbf{TP}}{\mathbf{TP + FP}} \text{ X } 100\% = \frac{\mathbf{225}}{\mathbf{225 + 22}} \text{ X } 100\% = 91.09\%$$

$$\text{Presisi Negatif} = \frac{\mathbf{TN}}{\mathbf{TN + FN}} \text{ X } 100\% = \frac{\mathbf{512}}{\mathbf{512 + 367}} \text{ X } 100\% = 58.25\%$$

$$\text{Recall Positif} = \frac{\mathbf{TP}}{\mathbf{TP + FN}} \text{ X } 100\% = \frac{\mathbf{225}}{\mathbf{225 + 367}} \text{ X } 100\% = 38\%$$

$$\text{Recall Negatif} = \frac{\mathbf{TN}}{\mathbf{TN + FP}} \text{ X } 100\% = \frac{\mathbf{512}}{\mathbf{512 + 22}} \text{ X } 100\% = 95.88\%$$

$$\textit{f1-score} \text{ Positif} = 2 \text{ X} \frac{\mathbf{presisi\ positif\ x\ recall\ positif}}{\mathbf{presisi\ positif + recall\ positif}} = 2 \text{ X} \frac{\mathbf{0.9109\ x\ 0.38}}{\mathbf{0.9109 + 0.38}} =$$

53.63%

$$\textit{f1-score} \text{ Positif} = 2 \text{ X} \frac{\mathbf{presisi\ negatif\ x\ recall\ negatif}}{\mathbf{presisi\ negatif + recall\ negatif}} = 2 \text{ X} \frac{\mathbf{0.5825\ x\ 0.9588}}{\mathbf{0.5825 + 0.9588}} = 72\%$$

Setelah didapati rasio dengan nilai akurasi terbaik, selanjutnya dilakukan *cross validation*. *Cross validation* dilakukan untuk mengetahui kinerja minimum dan maksimum yang didapat. Penelitian ini menggunakan 10-*folds cross validation* dimana nilai optimal didapatkan di k=11. Berikut tabel hasil dari 10-*folds cross validation*.

Tabel 4.13 Hasil 10-*folds Cross Validation*

n-fold	Akurasi	Presisi	*Recall*	*f1-score*
1	62%	72%	65%	60%
2	61%	71%	63%	57%
3	60%	71%	63%	57%
4	60%	71%	63%	57%
5	61%	71%	66%	60%
6	61%	72%	64%	58%
7	60%	72%	65%	58%
8	62%	72%	65%	60%
9	62%	72%	63%	58%
10	64%	74%	66%	62%

4.6 Interpretasi Hasil

Hasil analisis dari penelitian ini, telah dilakukan proses pengambilan data dengan menggunakan teknik *web scraping* yang didapatkan dari video YouTube Raymond Chin yang berjudul “2023: Menuju KEHANCURAN DUNIA” sebanyak 11.524 data komentar. Pengambilan data komentar tersebut menggunakan *tools* Google *Collaboratory* dimana bahasa pemrograman yang dipakai adalah bahasa pemrograman python.

Analisis didalam penelitian ini menggunakan 2 *tools* yaitu RapidMiner 10.0.0 dan Jupyter Notebook dengan bahasa pemrograman python. Tahapan tahapan yang dilakukan didalam penelitian ini diantaranya adalah *web scraping*, *text pre-processing*, pelabelan dataset dengan menggunakan kamus lexicon, klasifikasi data menggunakan 2 metode klasifikasi yaitu dengan Naïve Bayes dan K-Nearest Neighbor, pengujian *confusion matrix* dan visualisasi dimana visualisasi menggunakan *wordcloud*.

Pada proses pengambilan dataset menggunakan bahasa pemrograman python, proses menggunakan Google Colaboratory untuk memudahkan peneliti dalam pengambilan langsung dataset dari komentar YouTube Raymond Chin berjudul "2023: Menuju KEHANCURAN DUNIA" yang bertemakan resesi pada tahun 2023, karena dataset mengambil API-*key* YouTube dari *console-cloud* google. Dataset yang diambil tersebut kemudian disimpan didalam file csv. File csv tersebut berisikan 11.524 data komentar YouTube.

Proses pengambilan data tersebut menghasilkan file csv yang berisikan 4 kolom rincian yaitu *publishedAt* yaitu waktu saat pengguna melontarkan komentarnya pada video YouTube ini, *authorDisplayedName* yaitu nama dari akun yang berkomentar, *textDisplayed* yaitu isi dari komentar, dan yang terakhir *likeCount* yaitu jumlah pengguna yang menyukai komentar tersebut. Peneliti pada penelitian ini hanya akan menggunakan 2 kolom dari 4 kolom yang didapat yaitu *publishedAt* yaitu waktu komentar dan *textDisplayed* yaitu isi komentar. Kemudian komentar diurutkan dari yang paling awal hingga yang paling terbaru. Rentang waktu tersebut adalah dari tanggal 3 Oktober 2022 hingga 6 Februari 2023. Gambar grafik 4.12 dan 4.13 selanjutnya ditampilkan untuk memvisualisasikan data yang didapat dari rentang

waktu tersebut. Didapat pada bulan Oktober 2022 yang paling banyak menyita perhatian pengguna YouTube khususnya pengguna YouTube Indonesia dengan jumlah sekitar 10.606 komentar.

Selanjutnya data yang sudah didapatkan dari teknik *web scraping* tersebut dilakukan tahap *pre-processing*. Tahap *pre-processing* pada penelitian ini meliputi beberapa tahapan lagi yaitu seperti *case folding, cleaning, tokenize, normalize, stemming, dan stopword removal*. Dataset yang sudah dilakukan tahapan-tahapan *pre-processing* ini selanjutnya disebut dengan data komentar bersih yang akhirnya berjumlah 11.262 data komentar bersih.

Data yang sudah dibersihkan dan terstruktur selanjutnya dilakukan proses pelabelan yaitu melabelkan data menjadi 2 kelas sentimen yaitu kelas positif dan kelas nagatif. Pelabelan kelas didalam dataset ini menggunakan salah satu tahap pemodelan yaitu *lexicon based*. Kata-kata atau data komentar yang ada pada dataset dibandingkan dengan dokumen kamus *lexicon*. Data komentar tersebut akan dibandingkan dan diberikan nilai atau *score* sesuai dengan kamus *lexicon*. Jumlah dari *score* tersebut akan menentukan data komentar tersebut memiliki label kelas positif atau kelas negatif. Kamus *lexicon* yang digunakan bersumber dari Koto (2017) *Indonesian Sentiment* (Inset). Hasil dari perbandingan data komentar dengan dokumen kamus *lexicon* ini terdapat 8.072 komentar berlabel positif dan terdapat 3.607 komentar berlabel negatif. Dengan menggunakan bahasa pemrograman python, peneliti dapat menunjukkan hasil kata-kata yang sering diucapkan yang tervisualisasikan Digambar 4.22. Pada gambar tersebut 3 kata teratas yang sering muncul adalah kata Indonesia, dunia, dan uang. Kemudian peneliti juga memvisualisasikan *word frequency* atau frekuensi kata yang sering muncul dan *word cloud* pada masing-masing label kelas

positif dan kelas negatif yang ditunjukkan pada gambar 4.23 dan 4.24 untuk hasil label positif dan gambar 4.25 dan 4.26 untuk hasil label negatif. Data yang sudah diberikan label akan disimpan dalam file csv untuk proses selanjutnya.

Sebelum memasuki tahap *asses*, peneliti didalam penelitian ini membagi terlebih dahulu data uji dan data latih dari dataset yang sebelumnya sudah diproses. Peneliti membagi 3 rasio pembagian data latih dan data uji ini. Ketiga rasio tersebut adalah 70% data latih dengan 30% data uji, 80% data latih dengan 20% data uji, dan 90% data latih dengan 10% data uji berdasarkan penelitian sebelumnya dari Gormantara pada tahun 2020 (Gormantara 2020). Pembagian rasio ini dilakukan pada kedua model klasifikasi yaitu Naïve bayes dan K-Nearest Neighbor. Model klasifikasi K-Nearest Neighbor (KNN) setelah pembagian rasio tersebut kemudian akan dilakukan uji variasi nilai k. Nilai k terbaik dicari dari bantuan algoritma KNN, pembobotan TF-IDF dan 10-*fold cross validation*. Nilai k terbaik akan terlihat dari nilai *error* terkecil dengan variasi k yang akan diuji yaitu 10 nilai ganjil pertama dimulai dari k=1 sampai k=19.

Tahap terakhir yaitu tahap *asses* pada penelitian ini. Tahap ini dilakukan evaluasi dari 2 model klasifikasi yaitu Naïve Bayes dan K-Nearest Neighbor. Evaluasi tersebut berisikan nilai *confusion matrix*, nilai akurasi, nilai presisi, nilai *recall*, dan *f1-score* dari data uji pada masing masing model klasifikasi yaitu Naïve Bayes dan KNN. Kemudian metode KNN selanjutnya akan dilakukan 10-*fold cross validation* dalam pencarian nilai variasi k terbaik. Hasil nilai akurasi, presisi, *recall*, dan *f1-score* dari masing-masing metode klasifikasi ditunjukkan pada tabel 4.8 untuk metode Naïve Bayes dan tabel 4.11 untuk metode KNN. Berikut hasil rangkuman hasil nilai akurasi, presisi, *recall*, dan *f1-score*.

Tabel 4.14 Hasil Akurasi, Presisi, *Recall*, dan *f1-score* NBC dan KNN

Rasio	Naïve Bayes				K-Nearest Neighbor			
	Akurasi	Presisi	*Recall*	*f1*	Akurasi	Presisi	*Recall*	*f1*
70 : 30	69.43%	69.73%	68.8%	68.78%	63.24%	63.9%	63.71%	63.2%
80 : 20	68.18%	68.67%	67.65%	67.51%	63.06%	73.53%	64.57%	59.79%
90 : 10	68.23%	68.85%	67.62%	67.45%	65.45%	74.67%	66.94%	63.05%

Pada tabel diatas dapat terlihat masing-masing nilai akurasi, nilai presisi, nilai *recall*, dan nilai f1-*score* dari rasio yang sebelumnya sudah ditentukan oleh peneliti. Berikut definisi dari *confusion matrix* tersebut :

1. Akurasi

   Akurasi merupakan jumlah dari prediksi data yang benar dibagi oleh jumlah semua data. Dengan kata lain akurasi mengukur sejauh mana model dapat memprediksi dengan benar secara keseluruhan.

2. Presisi

   Presisi merupakan pengukuran sejauh mana hasil prediksi positif dari model sudah benar. Presisi menunjukkan berapa data dari prediksi positif yang benar akan dibagi dengan semua prediksi positif.

3. *Recall*

   Recall merupakan pengukuran sejauh mana indentifikasi dengan benar dari model untuk kasus positif aktual. *Recall* menunjukkan berapa banyak data dari kasus positif aktual yang berhasil diidentifikasi.

4. F1-*Score*

   F1-*Score* merupakan gabungan dari nilai presisi dan nilai *recall* dimana untuk memberikan gambaran keseluruhan dari kinerja model yang digunakan. Dapat disimpulkan f1-*score* merupakan nilai rata-rata dari presisi dan *recall*. Antara presisi dan *recall* f1-*score* dapat memberikan keseimbangan yang berguna ketika terdapat ketidakseimbangan kelas yang signifikan.

Selanjutnya pada metode K-Nearest Neighbor dilakukan *cross validation*. *Cross validation* dilakukan untuk mengetahui kinerja minimum dan maksimum yang didapat. Peneliti menggunakan 10-*folds cross validation* dimana nilai optimal didapatkan di k=11 dengan hasil 10-*folds cross validation* metode KNN ditunjukkan pada tabel 4.13.

Dari penelitian yang sudah dilakukan, peneliti mendapatkan bahwa model terbaik didalam penelitian ini adalah metode Naïve Bayes pada dataset komentar YouTube Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA" dengan topik resesi tahun 2023 berada pada rasio 70% : 30% data latih : data uji dengan nilai akurasi sebesar 69.43% dan akurasi terbaik untuk metode KNN pada penelitian ini ada di rasio 90% : 10% data latih : data uji sebesar 65.45%. Pada penelitian ini model *Naïve Bayes* lebih baik dibandingkan dengan metode KNN, dapat dilihat pada tabel 4.14 bahwa dari semua rasio nilai akurasi dan nilai rata-rata presisi dan *recall* (f1-*score*) dari metode *Naïve Bayes* lebih besar dibandingkan dengan metode KNN.

BAB 5

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Peneliti didalam penelitian ini telah berhasil menerapkan model klasifikasi teks untuk mengetahui gambaran sentimen masyarakat Indonesia pada media sosial YouTube melalui komentar video Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA" mengenai resesi 2023. Video tersebut diunggah di tanggal 3 Oktober 2022 dan peneliti melakukan *web scraping* pada tanggal 6 Februari 2023, yang berarti sudah 154 hari atau 5 bulan setelah video tersebut diunggah oleh *channel* YouTube Raymond Chin, peneliti mengambil dataset komentar video Youtube tersebut. Dari dataset tersebut peneliti mendapatkan label kelas menggunakan metode *lexicon based*, membuat visualisasi kata dengan *wordcloud*, dan mendapatkan nilai akurasi, presisi, *recall*, dan *f1-score* dari 2 metode klasifikasi yaitu Naïve Bayes dan K-Nearest Neighbor.

Berdasarkan hasil pembahasan klasifikasi teks data komentar video YouTube Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA" mengenai resesi 2023 menggunakan metode *lexicon base*, Naïve Bayes dan K-Nearest Neighbor, maka dapat diambil kesimpulan sebagai berikut :

a. Pada Metode Naïve Bayes didapatkan alpha 100 dengan menghasilkan hasil terbaik masing-masing nilai akurasi, presisi, *recall*, dan *f1-score* adalah sebesar 69.43%, 69.73%, 68.8% dan 68.78%. Sedangkan pada metode K-Nearest Neighbor didapatkan nilai k terbaik adalah 11 dengan menghasilkan nilai akurasi, presisi, *recall*, dan *f1-score* adalah sebesar

65.45%, 74.67%, 66.94% dan 63.05%. Jika dibandingkan maka model atau metode klasifikasi terbaik pada penelitian ini adalah metode Naïve Bayes yang dapat dilihat dari nilai akurasi dan nilai rata-rata presisi dan *recall* (nilai f1-*score*) tersebut.

b. Pengambilan data *web scraping* didapatkan total sebanyak 11.524 data komentar yang kemudian dilakukan pembersihan dan didapat hasil pembersihan menjadi 11.262 data komentar. Data komentar tersebut menghasilkan 8.072 komentar dengan sentimen positif dan 3.607 komentar dengan sentimen negatif, sehingga dapat disimpulkan bahwa sentimen pengguna YouTube khususnya pengguna YouTube Indonesia terhadap video YouTube Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA" mengenai resesi 2023 ini medapatkan atau menghasilkan lebih banyak sentimen positif. Frekuensi kemunculan kata-kata bersentimen positif terbanyak adalah kata Indonesia, dunia, negara, ekonomi, terima kasih, kaya, tuhan, dan solusi. Dari frekuensi kemunculan kata-kata tersebut peneliti pada penelitian ini dapat memberi kesimpulan bahwa banyak pengguna YouTube Indonesia berterima kasih kepada Raymond Chin karena telah mengedukasi masyarakat atau pengguna YouTube didalam video tersebut. Disampaikan dari beberapa data komentar pengguna YouTube yang menonton video ini, mereka mendapatkan ilmu baru baik dari ilmu ekonomi, perencanaan keuangan dalam menghadapi resesi hingga pengetahuan baru dan solusi dari informasi yang diberikan.

c. Dari kesimpulan *point* b diatas, maka peneliti dapat merekomendasikan video YouTube Raymond Chin yang berjudul "2023: Menuju KEHANCURAN DUNIA" baik sebagai acuan video dengan topik serupa yaitu resesi dengan penjelasan yang mudah dimengerti dan penyampaian ilmu yang mudah diterima oleh kebanyakan masyarakat khususnya pengguna YouTube Indonesia (penonton video ini).

5.2 Saran

Berdasarkan hasil penelitian pada penelitian ini, peneliti memiliki beberapa saran yang dapat menjadi masukan dan bahan pertimbangan untuk penelitian selanjutnya disarankan untuk:

a. Menggunakan cara atau metode lain misalnya seperti metode *Hierarchical Dirichlet Process* atau *Latent Semantic Analysis* untuk pemodelan topik. Kemudian juga disarankan untuk menggunakan kamus lain selain *lexicon Indonesian Sentiment* (InSet).
b. Menggunakan dan menambahkan *Emoticon Detection Sentiment Analysis* atau dapat diartikan penelitian selanjutnya dapat meneliti emoji yang terdapat pada data komentar YouTube.
c. Menambahkan analisis sentimen dengan data komentar didalam dataset menggunakan bahasa asing selain bahasa Indonesia. Atau dapat menambahkan bahasa daerah tertentu karena semua kalangan dapat mengakses media sosial khususnya dalam penelitian ini media sosialnya adalah YouTube.

d. Penelitian selanjutnya dapat menggunakan metode lain selain metode Naïve Bayes dan K-Nearest Neighbor, sehingga dapat dibandingkan dan menentukan akurasi terbaik dalam menganalisis sentimen komentar YouTube.

e. Penelitian ini hanya menganalisis komentar YouTube 1 video saja, penelitian selanjutnya dapat membuat suatu sistem informasi berbasis komputer untuk menganalisis sentimen suatu video dengan hanya memasukkan video tertentu dan menghasilkan analisis sentimen dari video tersebut

DAFTAR PUSTAKA

A. Yani, Dhita Deviacita, Helen Sasty Pratiwi, and Hafiz Muhardi. 2019. "Implementasi Web Scraping untuk Pengambilan Data pada Situs Marketplace." *Jurnal Sistem dan Teknologi Informasi (JUSTIN)* 7(4):257. doi: 10.26418/justin.v7i4.30930.

Alizah, Muhammad Dwison, Arifin Nugroho, Ummu Radiyah, and Windu Gata. 2020. "Sentimen Analisis Terkait Lockdown pada Sosial Media Twitter." *Indonesian Journal on Software Engineering (IJSE)* 6(2):223–29. doi: 10.31294/ijse.v6i2.8991.

Arsiana, Nena Febby. 2021. *Analisis Comment To Likes Ratio Youtube Pada 5 Youtuber Indonesia Dengan Subscriber Terbanyak*. *preprint*. Open Science Framework. doi: 10.31219/osf.io/srdtu.

Azhar. 2019. "Analisis Kinerja Algoritma Naive Bayes Dan K-Nearest Neighbor Pada Sentimen Analisis Dengan Pendekatan Lexicon Di Media Twitter." *Fakultas Sains Dan Teknologi Universitas Islam Negeri Syarif Hidayatullah Jakarta*.

Batubara, Hamdan Husein, and Dessy Noor Ariani. 2016. "Pemanfaatan Video sebagai Media Pembelajaran Matematika SD/MI." *Muallimuna : Jurnal Madrasah Ibtidaiyah* 2(1):47. doi: 10.31602/muallimuna.v2i1.741.

Blandina, Selena, Alvin Noor Fitrian, and Wulan Septiyani. 2020. "Strategi Menghindarkan Indonesia dari Ancaman Resesi Ekonomi di Masa Pandemi." *Efektor* 7(2):181–90. doi: 10.29407/e.v7i2.15043.

Buntoro, Ghulam Asrofi. 2017. "Analisis Sentimen Calon Gubernur DKI Jakarta 2017 Di Twitter." *INTEGER: Journal of Information Technology* 2(1). doi: 10.31284/j.integer.2017.v2i1.95.

Cahyanti, Dewi, Alifah Rahmayani, and Syafira Ainy Husniar. 2020. "Analisis performa metode Knn pada Dataset pasien pengidap Kanker Payudara." *Indonesian Journal of Data and Science* 1(2):39–43. doi: 10.33096/ijodas.v1i2.13.

Christianto, Maximillian, Justinus Andjarwirawan, and Alvin Tjondrowiguno. 2020. "Aplikasi Analisa Sentimen Pada Komentar Berbahasa Indonesia Dalam Objek Video di Website YouTube Menggunakan Metode Naïve Bayes Classifier." 8.

Deolika, Agatha, Kusrini Kusrini, and Emha Taufiq Luthfi. 2019. "ANALISIS PEMBOBOTAN KATA PADA KLASIFIKASI TEXT MINING." *JURNAL TEKNOLOGI INFORMASI* 3(2):179. doi: 10.36294/jurti.v3i2.1077.

Devita, Riri Nada, Heru Wahyu Herwanto, and Aji Prasetya Wibawa. 2018. "Perbandingan Kinerja Metode Naive Bayes dan K-Nearest Neighbor untuk

Klasifikasi Artikel Berbahasa indonesia." *Jurnal Teknologi Informasi dan Ilmu Komputer* 5(4):427. doi: 10.25126/jtiik.201854773.

Deviyanto, Akhmad, and Muhammad Didik Rohmad Wahyudi. 2018. "PENERAPAN ANALISIS SENTIMEN PADA PENGGUNA TWITTER MENGGUNAKAN METODE K-NEAREST NEIGHBOR." *JISKA (Jurnal Informatika Sunan Kalijaga)* 3(1):1. doi: 10.14421/jiska.2018.31-01.

Dong, Z., Guo Guo, S. Rajana, and B. Chen. 2020. "Understanding 21st Century Bordeaux Wines from Wine Reviews Using Naïve Bayes Classifier." doi: 10.3390/beverages6010005.

Fikri, Muhammad Ramadan, Rahmadya Trias Handayanto, and Dadan Irwan. 2022. "Web Scraping Situs Berita Menggunakan Bahasa Pemograman Python." *Journal of Students' Research in Computer Science* 3(1):123–36. doi: 10.31599/jsrcs.v3i1.1514.

Google Developers. 2023. "API Reference." *API Reference*. Retrieved (https://developers.google.com/youtube/v3/docs).

Gormantara, A. 2020. "Analisis Sentimen Terhadap New Normal Era Di Indonesia Pada Twitter Analisis Sentimen Terhadap New Normal Era Di Indonesia Pada Twitter Menggunakan Metode Support Vector Machine." (0–5).

Gupta, A., S. Gupta, and D. Singh. 2015. "A Systematic Review of Classification Techniques and Implementation of ID3 Decision Tree Algorithm." 144–52.

Herdhiyanto, Adhyksa. 2020. "Sentiment Analysis Menggunakan Naïve Bayes Classifier (NBC) Pada Tweet Tentang Zakat." *Institutional Repository UIN Syarif Hidayatullah* 122.

Khan, M. Laeeq. 2017. "Social Media Engagement: What Motivates User Participation and Consumption on YouTube?" *Computers in Human Behavior* 66:236–47. doi: 10.1016/j.chb.2016.09.024.

Kustiyahningsih, Yeni, and Nikmatus Syafa'ah. 2016. "SISTEM PENDUKUNG KEPUTUSAN UNTUK MENENTUKAN JURUSAN PADA SISWA SMA MENGGUNAKAN METODE KNN DAN SMART."

Larasati, Putu Karin Pradnya, Kashira Dwinda Kartika, Avivah Suci Rahayu, Putri Khairunisa, and I. Nyoman Larry Julianto. 2021. "Efektivitas Content Creator dalam Strategi Promosi di Era Digital (Effectiveness of Content Creators in Promotion Strategies in this Digital Age)." 1.

Liantoni, Febri. 2016. "Klasifikasi Daun Dengan Perbaikan Fitur Citra Menggunakan Metode K-Nearest Neighbor." *Jurnal ULTIMATICS* 7(2):98–104. doi: 10.31937/ti.v7i2.356.

Matulatuwa, Febrilien Matresya, Eko Sediyono, and Ade Iriani. 2017. "TEXT MINING DENGAN METODE LEXICON BASED UNTUK SENTIMENT

ANALYSIS PELAYANAN PT. POS INDONESIA MELALUI MEDIA SOSIAL TWITTER." *JURNAL MASYARAKAT INFORMATIKA INDONESIA* 2(3):13.

Mellyaningsih, Adinda. 2016. "Motif Subscriber Menonton Channel YouTube Raditya Dika." 4.

Miraza. 2019. "Seputar Resesi Dan Depresi." 30(2):11–13.

Mujianto, Haryadi. 2019. "PEMANFAATAN YOUTUBE SEBAGAI MEDIA AJAR DALAM MENINGKATKAN MINAT DAN MOTIVASI BELAJAR." 5(1).

Murphy, Lauren, Tosin Alliyu, Andrew Macvean, Mary Beth Kery, and Brad A. Myers. 2017. "Preliminary Analysis of REST API Style Guidelines."

Myers, and Stylos. 2016. "Improving API Usability." 59:62–69.

Nabila, Zulfa, Auliya Rahman Isnain, and Zaenal Abidin. 2021. "ANALISIS DATA MINING UNTUK CLUSTERING KASUS COVID-19 DI PROVINSI LAMPUNG DENGAN ALGORITMA K-MEANS." *Jurnal Teknologi dan Sistem Informasi* 2(2).

Nagamma, P., H. R. Pruthvi, K. K. Nisha, and N. H. Shwetha. 2015. "An Improved Sentiment Analysis of Online Movie Reviews Based on Clustering for Box-Office Prediction." Pp. 933–37 in *International Conference on Computing, Communication & Automation*. Greater Noida, India: IEEE.

Nurdin, and Dewi Astika. 2015. "PENERAPAN DATA MINING UNTUK MENGANALISIS PENJUALAN BARANG DENGAN MENGGUNAKAN METODE APRIORI PADA SUPERMARKET SEJAHTERA LHOKSEUMAWE." 6.

Pahlevi, Reza. 2022. "APJII: Penetrasi Internet Indonesia Capai 77,02% Pada 2022." *APJII: Penetrasi Internet Indonesia Capai 77,02% Pada 2022*. Retrieved (https://databoks.katadata.co.id/datapublish/2022/06/10/apjii-penetrasi-internet-indonesia-capai-7702-pada-2022).

Pambudi, Rilo, and Afif Afghohani. 2019. "Pengaruh Media Video Youtube Terhadap Prestasi Belajar Matematika Pada Siswa Kelas X SMK Negeri 2 Sukoharjo Tahun Ajaran 2017/2018." 28:175–82. doi: 10.32585/jp.v28i2.345.

Pires, Fernanda, Maria-Jose Masanet, and Carlos A. Scolari. 2021. "What Are Teens Doing with YouTube? Practices, Uses and Metaphors of the Most Popular Audio-Visual Platform." *Information, Communication & Society* 24(9):1175–91. doi: 10.1080/1369118X.2019.1672766.

Pratama, Yohanssen, Anthon Roberto Tampubolon, Liana Diantri Sianturi, Rifka Diana Manalu, and David Frietz Pangaribuan. 2019. "Implementation of

Sentiment Analysis on Twitter Using Naïve Bayes Algorithm to Know the People Responses to Debate of DKI Jakarta Governor Election." *Journal of Physics: Conference Series* 1175:012102. doi: 10.1088/1742-6596/1175/1/012102.

Pratiwi, Banu Putri, Ade Silvia Handayani, and Sarjana Sarjana. 2021. "PENGUKURAN KINERJA SISTEM KUALITAS UDARA DENGAN TEKNOLOGI WSN MENGGUNAKAN CONFUSION MATRIX." *Jurnal Informatika Upgris* 6(2). doi: 10.26877/jiu.v6i2.6552.

Puspita, Rani, and Agus Widodo. 2021. "Perbandingan Metode KNN, Decision Tree, dan Naïve Bayes Terhadap Analisis Sentimen Pengguna Layanan BPJS." *Jurnal Informatika Universitas Pamulang* 5(4):646. doi: 10.32493/informatika.v5i4.7622.

Rahmat, Brilian, Agum Agidtama Gafar, Nurul Fajriani, Umar Ramdani, Fitria Rihin Uyun, Yuwanda Purnamasari, and Natalis Ransi. 2017. "IMPLEMETASI K-MEANS CLUSTERING PADA RAPIDMINER UNTUK ANALISIS DAERAH RAWAN KECELAKAAN."

Ramadhan, Dery Anjas, and Erwin Budi Setiawan. 2019. "ANALISIS SENTIMEN PROGRAM ACARA DI SCTV PADA TWITTER MENGGUNAKAN METODE NAIVE BAYES DAN SUPPORT VECTOR MACHINE." 6:9742.

Rofqoh, Umi, Rizal Setya Perdana, and M. Ali Fauzi. 2017. "Analisis Sentimen Tingkat Kepuasan Pengguna Penyedia Layanan Telekomunikasi Seluler Indonesia Pada Twitter Dengan Metode Support Vector Machine dan Lexicon Based Features."

Rossum, Van, and Fred L. Drake. 2020. *Python Tutorial*. Vol. 620. 3.8.1. Centrum voor Wiskunde en Informatica.

Sahria, Yoga. 2020. "Implementasi Teknik Web Scraping pada Jurnal SINTA Untuk Analisis Topik Penelitian Kesehatan Indonesia."

Samsir, Samsir, Ambiyar Ambiyar, Unung Verawardina, Firman Edi, and Ronal Watrianthos. 2021. "Analisis Sentimen Pembelajaran Daring Pada Twitter di Masa Pandemi COVID-19 Menggunakan Metode Naïve Bayes." *JURNAL MEDIA INFORMATIKA BUDIDARMA* 5(1):157. doi: 10.30865/mib.v5i1.2580.

Saputra, Danandjaya, and Rizal Fathoni Aji. 2018. "ANALISIS PERBANDINGAN PERFORMA WEB SERVICE REST MENGGUNAKAN FRAMEWORK LARAVEL, DJANGO DAN RUBY ON RAILS UNTUK AKSES DATA DENGAN APLIKASI MOBILE." 2:6.

Sari, Yunita Ratna, Arby Sudewa, Diah Ayu Lestari, and Tri Ika Jaya. 2020. "Penerapan Algoritma K-Means Untuk Clustering Data Kemiskinan Provinsi

Banten Menggunakan Rapidminer." *CESS (Journal of Computer Engineering, System and Science)* 5(2):192. doi: 10.24114/cess.v5i2.18519.

Siringoringo, Rimbun, and Jamaludin Jamaludin. 2019. "Text Mining dan Klasterisasi Sentimen Pada Ulasan Produk Toko Online." *Jurnal Teknologi dan Ilmu Komputer Prima (JUTIKOMP)* 2(1):41–48. doi: 10.34012/jutikomp.v2i1.456.

Sunarmin, and Ahmad Junaidi. 2021. "Penentuan Strategi Bisnis Perusahaan Dalam Menghadapi Resesi Ekonomi." 8.

Susanti, Noor Aliyah, and Miftahul Walid. 2022. "KLASIFIKASI DATA TWEET UJARAN KEBENCIAN DI MEDIA SOSIAL MENGGUNAKAN NAIVE BAYES CLASSIFIER." 6(2).

Syaefulloh. 2020. "Web Scraping Menggunakan Python."

Tempola, Firman, Miftah Muhammad, and Amal Khairan. 2018. "Perbandingan Klasifikasi Antara KNN dan Naive Bayes pada Penentuan Status Gunung Berapi dengan K-Fold Cross Validation." *Jurnal Teknologi Informasi dan Ilmu Komputer* 5(5):577. doi: 10.25126/jtiik.201855983.

Viertola, Wilma. 2018. "To What Extent Does YouTube Marketing Influence the Consumer Behaviour of a Young Target Group?"

Wibawa, Aji Prasetya, Muhammad Guntur Aji Purnama, Muhammad Fathony Akbar, and Felix Andika Dwiyanto. 2018. "Metode-metode Klasifikasi." 3(1).

Yuanta, Friendha. 2019. "Pengembangan Media Video Pembelajaran Ilmu Pengetahuan Sosial pada Siswa Sekolah Dasar." 2:91–100.

Zhang, Chen, Liu, and Xi. 2020. "A Distributed Storage and Computation K-Nearest Neighbor Algorithm Based Cloud-Edge Computing for Cyber-Physical-Social Systems." doi: https://doi.org/10.1109/ACCESS.2020.2974764.

Zulqornain, Junda Alfiah, Indriati Indriati, and Pandu Adikara Putra. 2021. "Analisis Sentimen Tanggapan Masyarakat Aplikasi Tiktok Menggunakan Metode Naïve Bayes dan Categorial Propotional Difference (CPD)."

Zúñiga-López, Arturo, and Carlos Avilés-Cruz. 2020. "Digital Signal Processing Course on Jupyter–Python Notebook for Electronics Undergraduates." *Computer Applications in Engineering Education* 28(5):1045–57. doi: 10.1002/cae.22277.

LAMPIRAN

Source Code

Scrapping Data

```
#import library

import pandas as pd
from googleapiclient.discovery import build

def video_comments(video_id):

  # empty list for storing reply
  replies = []

  # creating youtube resource object
  youtube = build('youtube', 'v3', developerKey=api_key)

  # retrieve youtube video results
  video_response =
youtube.commentThreads().list(part='snippet,replies',
videoId=video_id).execute()

  # iterate video response
  while video_response:

    # extracting required info

    # from each result object
    for item in video_response['items']:

      # Extracting comments ()
      published =
item['snippet']['topLevelComment']['snippet']['publishedAt']

      user =
item['snippet']['topLevelComment']['snippet']['authorDisplayName']

      # Extracting comments
      comment =
item['snippet']['topLevelComment']['snippet']['textDisplay']

     likeCount =
item['snippet']['topLevelComment']['snippet']['likeCount']

      replies.append([published, user, comment, likeCount])

      # counting number of reply of comment
      replycount = item['snippet']['totalReplyCount']

      # if reply is there
      if replycount>0:
        # iterate through all reply
        for reply in item['replies']['comments']:

          # Extract reply
          published = reply['snippet']['publishedAt']
          user = reply['snippet']['authorDisplayName']
          repl = reply['snippet']['textDisplay']
          likeCount = reply['snippet']['likeCount']
```

```
            # Store reply is list

            #replies.append(reply)
            replies.append([published, user, repl, likeCount])

        # print comment with list of reply

        #print(comment, replies, end = '\n\n')

        # empty reply list

        #replies = []

      # Again repeat
      if 'nextPageToken' in video_response:
        video_response = youtube.commentThreads().list(
            part = 'snippet,replies',
            pageToken = video_response['nextPageToken'],
            videoId = video_id
          ).execute()
      else:
        break

  #endwhile
  return replies

# Isikan dengan api key
api_key = '---------------------------------'

# Enter video id

# Url video youtube = https://www.youtube.com/watch?v=Nhw6OWZjHVU

video_id = "Nhw6OWZjHVU" #isikan dengan kode / ID video

# Call function
comments = video_comments(video_id)

Comments
df = pd.DataFrame(comments, columns=['publishedAt',
'authorDisplayName', 'textDisplay', 'likeCount'])
df

#Menyimpan dalam format CSV
df.to_csv('youtube-comments.csv', index=False)
```

Text Preprocessing

```
# Mengambil Dataset
import pandas as pd
import numpy as np
data = pd.read_csv("youtube-comments-nobr.csv", encoding =
'unicode_escape')
data

# Memanggil Kolom yang di Butuhkan
df = data.filter(['textDisplay', 'publishedAt'])
df

#Mengurutkan sesuai waktu terbaru Data Komentar
df.sort_values(by='publishedAt')
df.loc[:,'publishedAt'] = pd.to_datetime(df.publishedAt)
df1 = df.set_index('publishedAt')
df_count = df1.resample('D').count()
df_count.sort_values(by='publishedAt')

#membuat grafik
import seaborn as sns
df_count.plot(y='textDisplay',figsize=(15,6))

# case folding
df['case_folding'] = df['textDisplay'].str.lower()

# cleaning
import string
import re
def cleaning(komentar):
    #remove ascii
    komentar = komentar.encode('ascii', 'replace').decode('ascii')

    #remove angka
    komentar = re.sub('[0-9]+', '', komentar)

    # remove mention, link, hashtag
    komentar = re.sub('<br.*?>(.+?)</br>',' ',komentar)
    komentar = ' '.join(re.sub("([@#][A-Za-z0-9]+)|(\w+:\/\/\S+)"," ",
komentar).split())
    komentar = re.sub('@[^\s]+','',komentar)

    #remove url
    komentar = re.sub(r'\w+:\/{2}[\d\w-]+(\.[\d\w-
]+)*(?:(?:\/[^\s/]*))*', '', komentar)

    #remove tanda baca
    komentar = re.sub(r'[^\w\d\s]+', '', komentar)

    #remove whitespace
    komentar = re.sub('\s+',' ',komentar)
```

```
    return komentar
df['cleaning'] = df['case_folding'].apply(cleaning)

# menghapus data duplikat
df.drop_duplicates(subset ="cleaning", keep = 'first', inplace = True)

# tokenize
from nltk.tokenize import word_tokenize
def word_tokenize_wrapper(cleaning):
    return word_tokenize(cleaning)
df['tokenize'] = df['cleaning'].apply(word_tokenize_wrapper)

# normalisasi
import nltk
normalizad_word = pd.read_excel(("normalisasi.xlsx"),
engine='openpyxl',)
normalizad_word_dict = {}
for index, row in normalizad_word.iterrows():
    if row[0] not in normalizad_word_dict:
        normalizad_word_dict[row[0]] = row[1]
def normalized_term(document):
    return [normalizad_word_dict[term] if term in normalizad_word_dict
else term for term in document]

df['normalize'] = df['tokenize'].apply(normalized_term)

# stopword removal
from nltk.corpus import stopwords
list_stopwords = stopwords.words('indonesian')

#tambahkan stopword manual
list_stopwords.extend(['tu', 'uf', 'deh','nak','amp','b','a', 'je','x',
                       'sih', 'dos', 'm', 'eh', 'tuh', 'hm', 'nya',
                       'ufufuf', 'lho', 'rm', 'nya', 'ufcc', 'ppv',
                       'dok', 'je', 'gb', 'pa', 'e', 'br', ])
sw = set(list_stopwords) - set (['jangan', 'jangankan', 'janganlah',
'tidak', 'tidakkah', 'tidaklah'])
def stopwords_removal(words):
    return [word for word in words if word not in sw]
df['stopword_removal'] = df['normalize'].apply(stopwords_removal)

# stemming
import Sastrawi
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
factory = StemmerFactory()
stemmer = factory.create_stemmer()
def stemmed_wrapper(term):
   return stemmer.stem(term)
df['stemming'] = df['stopword_removal'].apply(lambda x: [stemmer.stem(y) for y in x])
```

```
# menghapus tanda baca
def remove_punct(text):
    text  = " ".join([char for char in text if char not in
string.punctuation])
    return text
df['clean'] = df['stemming'].apply(lambda x: remove_punct(x))

#Fixing
import ast
def comment_fix(text):

   # nltk.download('stopwords')
    my_file = open("combined_stop_words.txt", "r")
    content = my_file.read()
    stop_words = content.split("\n")
    file_2  = open("combined_slang_words.txt", "r")
    content2 = file_2.read()
    slang_words = ast.literal_eval(content2)
    my_file.close()
    file_2.close()
    word_tokens = word_tokenize(text)
    for w in word_tokens:
        if w in slang_words.keys():
            word_tokens[word_tokens.index(w)] = slang_words[w]

    #filter using NLTK library append it to a string
    filtered_text = [w for w in word_tokens if not w in stop_words]
    filtered_text = []

    #looping through conditions
    for w in word_tokens:

        #check tokens against stop words , emoticons and punctuations
        if w not in stop_words and w not in string.punctuation:
            filtered_tweet.append(w.lower())
    return ' '.join(filtered_tweet)
df['Fixed'] = df['clean'].apply(lambda x: comment_fix(x))

#Menyimpan dalam file CSV
df.to_csv("preprocessing_result_final.csv")
```

Mambuat *Wordcloud*

```
#Import library
import pandas as pd
import numpy as np

#Import File yang digunakan
df = pd.read_csv(r"preprocessing_result_final.csv", encoding
="latin-1")
df

# Mengambil kolom yang Di butuhkan
df = df.filter(['Fixed', 'publishedAt'])
df

#Word Processing
from nltk.tokenize import word_tokenize

word_dict = {}
for i in range(0,len(df['Fixed'])):
    sentence = df['Fixed'][i]
    word_token = word_tokenize(str(sentence))
    for j in word_token:
        if j not in word_dict:
            word_dict[j] = 1
        else:
            word_dict[j] += 1

# Meng-import kamus Lexicon
negasi = ['bukan','tidak','ga','gk']
lexicon = pd.read_csv('modified_full_lexicon.csv')
# lexicon = lexicon.drop(lexicon[(lexicon['word'] ==
'bukan')
#                               |(lexicon['word'] ==
'tidak')
#                               |(lexicon['word'] ==
'ga')|(lexicon['word'] == 'gk') ].index,axis=0)
lexicon = lexicon.reset_index(drop=True)

lexicon_word = lexicon['word'].to_list()
lexicon_num_words = lexicon['number_of_words']

# Pemeriksaan kata yang tidak termasuk didalam Kamus Lexicon
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

ns_words = []
factory = StemmerFactory()
stemmer = factory.create_stemmer()
for word in word_dict.keys():
    if word not in lexicon_word:
        kata_dasar = stemmer.stem(word)
        if kata_dasar not in lexicon_word:
```

```
            ns_words.append(word)
len(ns_words)

len({k:v for (k,v) in word_dict.items() if ((k in
ns_words)&(v>3)) })

ns_words_list = {k:v for (k,v) in word_dict.items() if ((k
in ns_words)&(v>3))}

sort_orders = sorted(ns_words_list.items(), key=lambda x:
x[1], reverse=True)
sort_orders=sort_orders[0:20]
for i in sort_orders:
    print(i[0], i[1])

word_to_plot = df['Fixed'].copy()

def del_word(x,key_list):
    n = len(key_list)
    word_tokens = word_tokenize(x)
    new_x =''
    for word in word_tokens:
        if word not in key_list:
            new_x = new_x+word+' '
    return new_x

word_to_plot_1 = word_to_plot.apply(lambda x:
del_word(str(x),negasi))

text = word_to_plot_1.str.cat(others=None)

from wordcloud import WordCloud
import matplotlib.pyplot as plt

wordcloud = WordCloud(width = 800, height = 800,
background_color = 'white', max_words = 1000
                      , min_font_size = 20,
collocations=False).generate(text)
#plot the word cloud
fig = plt.figure(figsize = (8,8), facecolor = None)
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
#wordcloud.to_file('Wordcloud.png')

from wordcloud import WordCloud
import matplotlib.pyplot as plt

wordcloud = WordCloud(width = 800, height = 800,
background_color = 'black', max_words = 1000
                      , min_font_size = 20,
collocations=False).generate(text)
```

```
#plot the word cloud
fig = plt.figure(figsize = (8,8), facecolor = None)
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
#wordcloud.to_file('Wordcloud.png')
```

Klasifikasi Naïve Bayes

```
#Importing all the neccessary libraries
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from nltk.stem.porter import PorterStemmer
from nltk.corpus import stopwords
from nltk.stem.wordnet import WordNetLemmatizer
import pickle
import warnings
warnings.filterwarnings("ignore")
from sklearn import datasets, neighbors
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score
from collections import Counter
from sklearn.model_selection import cross_validate
#from sklearn import cross_validation
from matplotlib.colors import ListedColormap
from sklearn.metrics import confusion_matrix
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
import scikitplot.metrics as sciplot
from sklearn.metrics import accuracy_score
import sklearn.metrics as metrics
import math
from sklearn.decomposition import TruncatedSVD

#import dataset
data = pd.read_csv('Lexicon_Result_final.csv', encoding =
'unicode_escape')
data
```

```
final_data=data
final_data['SentimentPolarity'] =
final_data['sentiment'].apply(lambda x : 'Positive' if x > 0
else 'Negative')
final_data['Class_Labels'] =
final_data['SentimentPolarity'].apply(lambda x : 1 if x ==
'Positive' else 0)
final_data.head()

from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import TimeSeriesSplit,
GridSearchCV

def standardize(X_train_vectors, X_test_vectors):
    '''Function used to column standardize any given
matrix'''
    from sklearn.preprocessing import StandardScaler
    scalar = StandardScaler(with_mean=False)
    scalar.fit(X_train_vectors)
    X_train_vectors = scalar.transform(X_train_vectors)
    X_test_vectors = scalar.transform(X_test_vectors)
    print("The shape of the X_train_vectors is :
{}".format(X_train_vectors.shape))
    print("The shape of the X_test_vectors is :
{}".format(X_test_vectors.shape))
    return (X_train_vectors, X_test_vectors)

def performance(nb_classifier, vectorizationType, X_train,
y_train, X_test, y_test, optimal_alpha, mse): #MSE : Mean
Squared Loss
    '''Function to measure the various performance metrics
for a given model.'''
    print("\n'''PERFORMANCE EVALUATION'''")
    print("\n\nDetailed report for the {}
Vectorization.".format(vectorizationType))

    #Predict the labels for the test set.
    y_pred = nb_classifier.predict(X_test)

    #Evaluate the accuracy of the model on test set
    test_accuracy = accuracy_score(y_test, y_pred,
normalize=True) * 100
    points = accuracy_score(y_test, y_pred, normalize=False)
    print('\nThe number of accurate predictions out of {}
data points on unseen data is {}'.format(X_test.shape[0],
points))
    print('Accuracy of the {} model on unseen data is {}
%'.format(vectorizationType, np.round(test_accuracy,2)))

    #Get the precision, recall and F1 score for this model.
    print("Precision of the {} model on unseen data is
{}".format(vectorizationType,
```

```
np.round(metrics.precision_score(y_test ,y_pred,
average='macro'),4)))
    print("Recall of the {} model on unseen data is
{}".format(vectorizationType,
np.round(metrics.recall_score(y_test ,y_pred,
average='macro'),4)))
    print("F1 score of the {} model on unseen data is
{}".format(vectorizationType,
np.round(metrics.f1_score(y_test ,y_pred,
average='macro'),4)))

    #Classification Report
    print ('\nClasification report for {} model :
\n'.format(vectorizationType))
    print(metrics.classification_report(y_test,y_pred))

    #Inference
    print("\nOf all the reviews that the model has predicted
to be positive, {}% of them are actually
positive.".format(np.round(metrics.precision_score(y_test
,y_pred)*100,2)))
    print("Of all the reviews that are actually positive,
the model has predicted {}% of them to be
positive.".format(np.round(metrics.recall_score(y_test
,y_pred)*100,2)))

    #Save the below list for later use to display model
information
    info_model_NB = [vectorizationType, optimal_alpha,
np.round(np.array(mse).mean(),4), np.round(1-
metrics.accuracy_score(y_test, y_pred),4),
np.round(metrics.f1_score(y_test ,y_pred),4), points]
    with open('info_model_NB.txt', 'a') as filehandle:
        filehandle.writelines("%s " % iterator for iterator
in info_model_NB)
        filehandle.writelines("\n")

    #Get the confusion matrix for the running model
    print("\nFind below the confusion matrix for {}
model.".format(vectorizationType))
    sciplot.plot_confusion_matrix(y_test ,y_pred)

    #Free memory allocations
    del(X_train, y_train, X_test, y_test, vectorizationType,
y_pred, nb_classifier)

def get_GridSearchCV_estimator(vectorizationType, X_train,
y_train, X_test, y_test):
    '''This function will determine the best hyperparameters
using TimeSeriesSplit CV and Grid Search, using 10 fold
cross validation. '''
    from sklearn.model_selection import TimeSeriesSplit
```

```
    alphas = np.logspace(0, 2, 20)
    tuned_parameters = [{'alpha': alphas}]
    n_folds = 10
    model = MultinomialNB()
    gsearch_cv = GridSearchCV(estimator=model,
param_grid=tuned_parameters, cv=10, scoring='f1', n_jobs=6)
    gsearch_cv.fit(X_train, y_train)
    print("\nGridSearchCV completed for {}
model!".format(vectorizationType))
    print("Best estimator for {} model :
".format(vectorizationType), gsearch_cv.best_estimator_)
    print("Best Score for {} model :
".format(vectorizationType), gsearch_cv.best_score_)
    return gsearch_cv

def plot_errors(gsearch_cv):
    '''This function is used to plot the curve for mean
squared errors vs alpha values'''
    #Get cross validation scores. Here we obtain the alpha
values and their corresponding mean test scores.
    cv_result = gsearch_cv.cv_results_
    mts = cv_result["mean_test_score"]        #list that
will hold the mean of cross validation accuracy scores for
each alpha
    alphas = cv_result["params"]

    alpha_values = []                         #list that
will hold all the alpha values that the grid search cross
validator tried.
    for i in range(0,len(alphas)):
        alpha_values.append(alphas[i]["alpha"])

    #Changing accuracy to mean squared error. **error = 1 -
accuracy ; error = Cross Validation Errors, accuracy = Cross
Validation accuracy
    mse = [1 - x for x in mts]

    #Determining best alpha from errors. 'alpha' will be
best for the lowest value for error
    optimal_alpha = alpha_values[mse.index(min(mse))]
#Laplace smoothing
    print('The optimal value of alpha is :
{}'.format(optimal_alpha))

    #Plot error vs alpha values
    plt.figure(figsize=(35,8))
    plt.plot(alpha_values , mse, color='green',
linestyle='dashed', linewidth=2, marker='o',
markerfacecolor='red', markersize=10)
    for xy in zip(alpha_values, np.round(mse,3)):
        plt.annotate('(%s, %s)' % xy, xy=xy,
textcoords='data')
```

```
    plt.title('Plot for Errors vs Alpha Values')
    plt.xlabel('Values of Alpha')
    plt.ylabel('Errors')
    plt.show()

    return (optimal_alpha,mse)

def naive_bayes_algorithm(X_train, y_train, X_test, y_test,
vectorizationType, vectorizer_object):
    '''This function splits the dataset into training set
and test sets. The test data remains untouched.
    A time series 10 fold cross validation is performed on
the train data and the value of optimal alpha is calculated.
    The dataset is then trained with this value of optimal
alpha.
    Finally the Naive Bayes model is used to predict its
accuracy on the future unseen test set.'''

    #Perform 10-fold cross validation on the train set
    print("Starting Cross Validation steps...")
    gsearch_cv =
get_GridSearchCV_estimator(vectorizationType, X_train,
y_train, X_test, y_test)

    #Plot the graphical representation of the mean squared
error vs the alpha values obtained during cross validation.
    optimal_alpha, mse = plot_errors(gsearch_cv)

    #Initialize the Naive Bayes constructor using alpha =
optimal_alpha
    nb_classifier = gsearch_cv.best_estimator_

    #Fit the model to the train set using optimal alpha
    nb_classifier.fit(X_train, y_train)

    #Evaluate the model's performance
    performance(nb_classifier, vectorizationType, X_train,
y_train, X_test, y_test, optimal_alpha, mse)

    '''TF-IDF model creation using text reviews. HTML tags
and punctuations are removed. All stopwords are
preserved.'''

sampled_dataset = final_data

#Split the data set into train and test sets. The test set
should be unseen. Time Based Splitting Step 2.
#The top old 70% data will get into the train set. The
latest 30% data will get into the test set.
X = sampled_dataset['komentar'].values.astype('U')
y = sampled_dataset['Class_Labels']
split = math.floor(0.7*len(X))
```

```
X_train = X[0:split,] ; y_train = y[0:split,]
X_test = X[split:,] ; y_test = y[split:,]

#Initializing the TF-IDF contructor
tf_idf_object =
TfidfVectorizer(ngram_range=(1,1)).fit(X_train)

#Creating the BOW matrix from cleaned data corpus. Only
'not' is preserved from stopwords. This is done for both
train and test Vectors.
print("\nCreating the TFIDF vectors using the cleaned
corpus")
X_train_vectors = tf_idf_object.transform(X_train)
X_test_vectors = tf_idf_object.transform(X_test)

#Colum Standardization of the TF-IDF vector created using
cleaned data. This is done for both train and test Vectors.
X_train_vectors, X_test_vectors =
standardize(X_train_vectors, X_test_vectors)

#Free memory allocations.
del(sampled_dataset, X, y)

#Fitting the Naive Bayes to the BOW model
naive_bayes_algorithm(X_train_vectors, y_train,
X_test_vectors, y_test, "TF-IDF", tf_idf_object)

'''TF-IDF model creation using text reviews. HTML tags and
punctuations are removed. All stopwords are preserved.'''

sampled_dataset = final_data

#Split the data set into train and test sets. The test set
should be unseen. Time Based Splitting Step 2.
#The top old 80% data will get into the train set. The
latest 20% data will get into the test set.
X = sampled_dataset['komentar'].values.astype('U')
y = sampled_dataset['Class_Labels']
split = math.floor(0.8*len(X))
X_train = X[0:split,] ; y_train = y[0:split,]
X_test = X[split:,] ; y_test = y[split:,]

#Initializing the TF-IDF contructor
tf_idf_object =
TfidfVectorizer(ngram_range=(1,1)).fit(X_train)

#Creating the BOW matrix from cleaned data corpus. Only
'not' is preserved from stopwords. This is done for both
train and test Vectors.
print("\nCreating the TFIDF vectors using the cleaned
corpus")
X_train_vectors = tf_idf_object.transform(X_train)
```

```
X_test_vectors = tf_idf_object.transform(X_test)

#Colum Standardization of the TF-IDF vector created using
cleaned data. This is done for both train and test Vectors.
X_train_vectors, X_test_vectors =
standardize(X_train_vectors, X_test_vectors)

#Free memory allocations.
del(sampled_dataset, X, y)

#Fitting the Naive Bayes to the BOW model
naive_bayes_algorithm(X_train_vectors, y_train,
X_test_vectors, y_test, "TF-IDF", tf_idf_object)

'''TF-IDF model creation using text reviews. HTML tags and
punctuations are removed. All stopwords are preserved.'''

sampled_dataset = final_data

#Split the data set into train and test sets. The test set
should be unseen. Time Based Splitting Step 2.
#The top old 90% data will get into the train set. The
latest 10% data will get into the test set.
X = sampled_dataset['komentar'].values.astype('U')
y = sampled_dataset['Class_Labels']
split = math.floor(0.9*len(X))
X_train = X[0:split,] ; y_train = y[0:split,]
X_test = X[split:,] ; y_test = y[split:,]

#Initializing the TF-IDF contructor
tf_idf_object =
TfidfVectorizer(ngram_range=(1,1)).fit(X_train)

#Creating the BOW matrix from cleaned data corpus. Only
'not' is preserved from stopwords. This is done for both
train and test Vectors.
print("\nCreating the TFIDF vectors using the cleaned
corpus")
X_train_vectors = tf_idf_object.transform(X_train)
X_test_vectors = tf_idf_object.transform(X_test)

#Colum Standardization of the TF-IDF vector created using
cleaned data. This is done for both train and test Vectors.
X_train_vectors, X_test_vectors =
standardize(X_train_vectors, X_test_vectors)

#Free memory allocations.
del(sampled_dataset, X, y)

#Fitting the Naive Bayes to the BOW model
naive_bayes_algorithm(X_train_vectors, y_train,
X_test_vectors, y_test, "TF-IDF", tf_idf_object)
```

```
# splitting data
sampled_dataset = final_data

#Split the data set into train and test sets. The test set
should be unseen. Time Based Splitting Step 2.
#The top old 80% data will get into the train set. The
latest 20% data will get into the test set.
X = sampled_dataset['komentar'].values.astype('U')
y = sampled_dataset['Class_Labels']
split = math.floor(0.7*len(X))
X_train = X[0:split,] ; y_train = y[0:split,]
X_test = X[split:,] ; y_test = y[split:,]

#Initializing the TF-IDF contructor
tf_idf_object =
TfidfVectorizer(ngram_range=(1,1)).fit(X_train)

#Creating the BOW matrix from cleaned data corpus. Only
'not' is preserved from stopwords. This is done for both
train and test Vectors.
print("\nCreating the TFIDF vectors using the cleaned
corpus")
X_train_vectors = tf_idf_object.transform(X_train)
X_test_vectors = tf_idf_object.transform(X_test)

#Colum Standardization of the TF-IDF vector created using
cleaned data. This is done for both train and test Vectors.
X_train_vectors, X_test_vectors =
standardize(X_train_vectors, X_test_vectors)

#Free memory allocations.
del(sampled_dataset, X, y)

# cross validation
from sklearn.model_selection import KFold
from sklearn.metrics import classification_report
from sklearn.preprocessing import StandardScaler
sampled_dataset = final_data
X = sampled_dataset['komentar'].values.astype('U')
y = sampled_dataset['Class_Labels']

kf = KFold(n_splits=10)
def cross_val(estimator):
    acc = []
    pcs = []
    rec = []
    f1 = []

    for train_index, test_index in kf.split(X, y):
        X_train, X_test = X[train_index], X[test_index]
        y_train, y_test = y[train_index], y[test_index]
```

```
        tf_idf_object =
TfidfVectorizer(ngram_range=(1,1)).fit(X_train)
        X_train_vec = tf_idf_object.transform(X_train)
        X_test_vec = tf_idf_object.transform(X_test)

        scalar = StandardScaler(with_mean=False)
        scalar.fit(X_train_vec)
        X_train_vectors = scalar.transform(X_train_vec)
        X_test_vectors = scalar.transform(X_test_vec)

        model = estimator.fit(X_train_vectors, y_train)
        y_pred = model.predict(X_test_vectors)

        acc.append(accuracy_score(y_test, y_pred))
        pcs.append(precision_score(y_test, y_pred,
average='macro', zero_division=0))
        rec.append(recall_score(y_test, y_pred,
average='macro', zero_division=0))
        f1.append(f1_score(y_test, y_pred, average='macro',
zero_division=0))

        print(classification_report(y_test, y_pred,
zero_division=0))
        print(f'confusion matrix:\n
{confusion_matrix(y_test, y_pred)}')

print('=====================================================
\n')

    print(f'average akurasi: {np.mean(acc)}')
    print(f'average presisi: {np.mean(pcs)}')
    print(f'average recall: {np.mean(rec)}')
    print(f'average f1-score: {np.mean(f1)}')

from sklearn.naive_bayes import MultinomialNB
nb = MultinomialNB(alpha=100, fit_prior=True,
class_prior=None)
cross_val(nb)
```

Klasifikasi K-Nearest Neighbor

```
#Importing all the neccessary libraries
%matplotlib inline
import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.feature_extraction.text import TfidfTransformer
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from nltk.stem.porter import PorterStemmer
from nltk.corpus import stopwords
from nltk.stem.wordnet import WordNetLemmatizer
import pickle
import warnings
warnings.filterwarnings("ignore")
import numpy as np
from sklearn import datasets, neighbors
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score
from collections import Counter
from sklearn.model_selection import cross_validate
#from sklearn import cross_validation
from matplotlib.colors import ListedColormap
from sklearn.metrics import confusion_matrix
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score
import sklearn.metrics as metrics
import math
from sklearn.decomposition import TruncatedSVD

#import file
df = pd.read_csv('Lexicon_Result_final.csv', encoding =
'unicode_escape')
df

final_data=df
final_data['SentimentPolarity'] =
final_data['sentiment'].apply(lambda x : 'Positive' if x > 0
else 'Negative')
final_data['Class_Labels'] =
final_data['SentimentPolarity'].apply(lambda x : 1 if x ==
'Positive' else 0)
final_data.head()

#Display distribution of Postive and Negative reviews in a
bar graph
final_data["Class_Labels"].value_counts().plot(kind='bar',co
lor=['green','red'],title='Distribution Of Positive and
Negative reviews',figsize=(5,5))

#Classifier Data
```

```
def knn_algorithm(X_train, y_train, X_test, y_test,
vectorizationType):
    '''This function splits the dataset into training set
and test sets. The test data remains untouched.
    A 10 fold cross validation is performed on the train
data and the value of optimal K is calculated.
    The dataset is then trained with this value of optimal
k.
    Finally the knn model is used to predict its accuracy on
the future unseen test set.'''

    X_train = X_train ; y_train = y_train #Train dataframe
    X_test = X_test ; y_test = y_test #Test Dataframe

    #algorithms = ['brute','kd_tree']
    algorithms = ['brute']

    for algo in algorithms:

        print("\nStarting Cross Validation steps for {}
model.".format(vectorizationType, algo.upper()))

        #Creating an odd number list of different K values
for KNN.
        k_values = list(np.arange(1,20,2))

        #Create an empty list that will hold the mean of
cross validation accuracy scores for each value of k in the
CV step.
        cross_val_scores = []

        if algo == 'kd_tree':
            svd = TruncatedSVD(n_components = 100)
            X_train = svd.fit_transform(X_train)
            X_test = svd.fit_transform(X_test)

        #Perform 10-fold cross validation on the train set
        for k in k_values:
            knn_classifier =
KNeighborsClassifier(n_neighbors=k, weights='distance',
algorithm=algo, p=2, metric='minkowski', n_jobs=6)
            accuracies = cross_val_score(knn_classifier,
X_train, y_train, cv=10, scoring='accuracy')
            cross_val_scores.append(accuracies.mean())
            #print("Cross validation completed using k =
{}".format(k))

        #Changing accuracy to error. **error = 1 - accuracy
        errors = [1 - x for x in cross_val_scores]

        #Determining best k from errors. K will be best for
the lowest value for error.
```

```
        optimal_k = k_values[errors.index(min(errors))]
        print('\nThe optimal number of neighbors is :
{}'.format(optimal_k))

        #Plot errors vs k values
        plt.figure(figsize=(12,6))
        plt.plot(k_values , errors, color='green',
linestyle='dashed', linewidth=2, marker='o',
markerfacecolor='red', markersize=10)
        for xy in zip(k_values, np.round(errors,3)):
            plt.annotate('(%s, %s)' % xy, xy=xy,
textcoords='data')
        plt.title('Plot for Errors vs K Values')
        plt.xlabel('Number of Neighbors
K'.format(algo.upper()))
        plt.ylabel('Errors')
        plt.show()

        print("The error for each k value:
{}".format(np.round(errors,3)))

        '''Train the model using the optimal value of k
found from the previous step and evaluate it's accuracy on
the test set(unseen data).'''

        #Initialize the KNN model, where k = optimal_k
        knn_classifier =
KNeighborsClassifier(n_neighbors=optimal_k,
weights='distance', algorithm='kd_tree', p=2,
metric='minkowski', n_jobs=6)

        #Fit the model to the train set
        knn_classifier.fit(X_train, y_train)

        #Predict the labels for the test set.
        y_pred = knn_classifier.predict(X_test)

        '''PERFORMANCE EVALUATION'''

        print("\n'''PERFORMANCE EVALUATION FOR {}
model'''".format(vectorizationType))

        print("\n\nDetailed report for the {}
Vectorization:".format(vectorizationType))

        #Evaluate the accuracy of the model on test set
        test_accuracy = accuracy_score(y_test, y_pred,
normalize=True) * 100
        points = accuracy_score(y_test, y_pred,
normalize=False)
```

```
        print('The number of accurate predictions out of {}
data points on unseen data for K = {} is
{}'.format(X_test_vectors.shape[0],optimal_k, points))
        print('\nAccuracy of the KNN model on unseen data
for K = {} is {} %'.format(optimal_k,
np.round(test_accuracy,2)))

        #Get the precision, recall and F1 score for this
model.
        print("Precision of the KNN model on unseen data for
K = {} is {}".format(optimal_k,
np.round(metrics.precision_score(y_test ,y_pred,
average='macro'),4)))
        print("Recall of the KNN model on unseen data for K
= {} is {}".format(optimal_k,
np.round(metrics.recall_score(y_test ,y_pred,
average='macro'),4)))
        print("F1 score of the KNN model on unseen data for
K = {} is {}".format(optimal_k,
np.round(metrics.f1_score(y_test ,y_pred,
average='macro'),4)))

        #Classification Report
        print ('\nClasification report for {} model:
\n'.format(vectorizationType))
        print(metrics.classification_report(y_test,y_pred))

        #Inference
        print("\nOf all the reviews that the model has
predicted to be positive, {}% of them are actually
positive.".format(np.round(metrics.precision_score(y_test
,y_pred)*100,2)))
        print("Of all the reviews that are actually
positive, the model has predicted {}% of them to be
positive.".format(np.round(metrics.recall_score(y_test
,y_pred)*100,2)))

        #Get the confusion matrix for the running model
        print("\nFind below the confusion matrix for {}
model.".format(vectorizationType))
        sciplot.plot_confusion_matrix(y_test ,y_pred)

        #Save the below list for later use to display model
information
        info_model_KNN = [vectorizationType, optimal_k,
np.round(np.array(errors).mean(),4), np.round(1-
metrics.accuracy_score(y_test, y_pred),4),
np.round(metrics.f1_score(y_test ,y_pred),4)]
        with open('info_model_KNN.txt', 'a') as filehandle:
            filehandle.writelines("%s " % iterator for
iterator in info_model_KNN)
            filehandle.writelines("\n")
```

```
        #Freeing memory allocations
        del(X_train, y_train, X_test, y_test, y_pred,
knn_classifier)

        '''TF-IDF model creation using text reviews. HTML
tags and punctuations are removed. All stopwords are
preserved.'''

import scikitplot.metrics as sciplot
sampled_dataset = final_data

#Split the data set into train and test sets. The test set
should be unseen. Time Based Splitting Step 2.
#The top old 90% data will get into the train set. The
latest 10% data will get into the test set.
X = sampled_dataset['komentar'].values.astype('U')
y = sampled_dataset['Class_Labels']
#split = math.floor(0.8*len(X))
split = 10136
X_train = X[0:split,] ; y_train = y[0:split,]
X_test = X[split:11262,] ; y_test = y[split:11262,]

#Initializing the TF-IDF contructor
tf_idf_object =
TfidfVectorizer(ngram_range=(1,1)).fit(X_train)

#Creating the BOW matrix from cleaned data corpus. Only
'not' is preserved from stopwords. This is done for both
train and test Vectors.
print("\nCreating the BOW vectors using the cleaned corpus")
X_train_vec = tf_idf_object.transform(X_train)
X_test_vec = tf_idf_object.transform(X_test)

#Colum Standardization of the TF-IDF vector created using
cleaned data. This is done for both train and test Vectors.
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler(with_mean=False)
X_train_scaler = scaler.fit(X_train_vec)
X_train_vectors = X_train_scaler.transform(X_train_vec)
X_test_vectors = X_train_scaler.transform(X_test_vec)
print("The shape of the X_train_vectors is :
{}".format(X_train_vectors.shape))
print("The shape of the X_test_vectors is :
{}".format(X_test_vectors.shape))

#Free memory allocations.
del(sampled_dataset, X, y, X_train, X_test)

#Fitting the KNN to the TF-IDF model
knn_algorithm(X_train_vectors, y_train, X_test_vectors,
y_test, "TF-IDF")
```

```
'''TF-IDF model creation using text reviews. HTML tags and
punctuations are removed. All stopwords are preserved.'''

sampled_dataset = final_data

#Split the data set into train and test sets. The test set
should be unseen. Time Based Splitting Step 2.
#The top old 80% data will get into the train set. The
latest 20% data will get into the test set.
X = sampled_dataset['komentar'].values.astype('U')
y = sampled_dataset['Class_Labels']
#split = math.floor(0.8*len(X))
split = 9010
X_train = X[0:split,] ; y_train = y[0:split,]
X_test = X[split:11262,] ; y_test = y[split:11262,]

#Initializing the TF-IDF contructor
tf_idf_object =
TfidfVectorizer(ngram_range=(1,1)).fit(X_train)

#Creating the BOW matrix from cleaned data corpus. Only
'not' is preserved from stopwords. This is done for both
train and test Vectors.
print("\nCreating the BOW vectors using the cleaned corpus")
X_train_vectors = tf_idf_object.transform(X_train)
X_test_vectors = tf_idf_object.transform(X_test)

#Colum Standardization of the TF-IDF vector created using
cleaned data. This is done for both train and test Vectors.
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler(with_mean=False)
X_train_scaler = scaler.fit(X_train_vectors)
X_train_vectors = X_train_scaler.transform(X_train_vectors)
X_test_vectors = X_train_scaler.transform(X_test_vectors)
print("The shape of the X_train_vectors is :
{}".format(X_train_vectors.shape))
print("The shape of the X_test_vectors is :
{}".format(X_test_vectors.shape))

#Free memory allocations.
del(sampled_dataset, X, y, X_train, X_test)

#Fitting the KNN to the TF-IDF model
knn_algorithm(X_train_vectors, y_train, X_test_vectors,
y_test, "TF-IDF")

'''TF-IDF model creation using text reviews. HTML tags and
punctuations are removed. All stopwords are preserved.'''

sampled_dataset = final_data
```

```
#Split the data set into train and test sets. The test set
should be unseen. Time Based Splitting Step 2.
#The top old 70% data will get into the train set. The
latest 30% data will get into the test set.
X = sampled_dataset['komentar'].values.astype('U')
y = sampled_dataset['Class_Labels']
#split = math.floor(0.8*len(X))
split = 7883
X_train = X[0:split,] ; y_train = y[0:split,]
X_test = X[split:12647,] ; y_test = y[split:12647,]

#Initializing the TF-IDF contructor
tf_idf_object =
TfidfVectorizer(ngram_range=(1,1)).fit(X_train)

#Creating the BOW matrix from cleaned data corpus. Only
'not' is preserved from stopwords. This is done for both
train and test Vectors.
print("\nCreating the BOW vectors using the cleaned corpus")
X_train_vectors = tf_idf_object.transform(X_train)
X_test_vectors = tf_idf_object.transform(X_test)

#Colum Standardization of the TF-IDF vector created using
cleaned data. This is done for both train and test Vectors.
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler(with_mean=False)
X_train_scaler = scaler.fit(X_train_vectors)
X_train_vectors = X_train_scaler.transform(X_train_vectors)
X_test_vectors = X_train_scaler.transform(X_test_vectors)
print("The shape of the X_train_vectors is :
{}".format(X_train_vectors.shape))
print("The shape of the X_test_vectors is :
{}".format(X_test_vectors.shape))

#Free memory allocations.
del(sampled_dataset, X, y, X_train, X_test)

#Fitting the KNN to the TF-IDF model
knn_algorithm(X_train_vectors, y_train, X_test_vectors,
y_test, "TF-IDF")

# cross validation
from sklearn.model_selection import KFold
from sklearn.metrics import classification_report
from sklearn.preprocessing import StandardScaler
import scikitplot.metrics as sciplot
sampled_dataset = final_data

X = sampled_dataset['komentar'].values.astype('U')
y = sampled_dataset['Class_Labels']
#split = math.floor(0.9*len(X))
split = 10136
```

```
kf = KFold(n_splits=10)
def cross_val(estimator):
    acc = []
    pcs = []
    rec = []
    f1 = []

    for train_index, test_index in kf.split(X, y):
        X_train, X_test = X[train_index], X[test_index]
        y_train, y_test = y[train_index], y[test_index]

        tf_idf_object =
TfidfVectorizer(ngram_range=(1,1)).fit(X_train)
        X_train_vec = tf_idf_object.transform(X_train)
        X_test_vec = tf_idf_object.transform(X_test)

        scalar = StandardScaler(with_mean=False)
        scalar.fit(X_train_vec)
        X_train_vectors = scalar.transform(X_train_vec)
        X_test_vectors = scalar.transform(X_test_vec)

        model = estimator.fit(X_train_vectors, y_train)
        y_pred = model.predict(X_test_vectors)

        acc.append(accuracy_score(y_test, y_pred))
        pcs.append(precision_score(y_test, y_pred,
average='macro', zero_division=0))
        rec.append(recall_score(y_test, y_pred,
average='macro', zero_division=0))
        f1.append(f1_score(y_test, y_pred, average='macro',
zero_division=0))

        print(classification_report(y_test, y_pred,
zero_division=0))
        print(f'confusion matrix:\n
{confusion_matrix(y_test, y_pred)}')

print('=====================================================
\n')


    print(f'average akurasi: {np.mean(acc)}')
    print(f'average presisi: {np.mean(pcs)}')
    print(f'average recall: {np.mean(rec)}')
    print(f'average f1-score: {np.mean(f1)}')

from sklearn.naive_bayes import MultinomialNB
knn = KNeighborsClassifier(n_neighbors=11,
algorithm='kd_tree', leaf_size=30, p=2, metric='minkowski',
metric_params=None, n_jobs=None)
cross_val(knn)
```