ANALISIS SENTIMEN PUBLIK TERHADAP KEBIJAKAN PEMBERLAKUAN PEMBATASAN KEGIATAN MASYARAKAT SKALA MIKRO MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE STUDI KASUS TWITTER

TUGAS AKHIR

Program Studi
S1 Sistem Informasi

Oleh:
Renas Madya Pradhana
17410100180

FAKULTAS TEKNOLOGI DAN INFORMATIKA
UNIVERSITAS DINAMIKA
2021

ANALISIS SENTIMEN PUBLIK TERHADAP KEBIJAKAN PEMBERLAKUAN PEMBATASAN KEGIATAN MASYARAKAT SKALA MIKRO MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE STUDI KASUS TWITTER

TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk menyelesaikan
Program Sarjana Komputer

Oleh:

Nama	**: Renas Madya Pradhana**
NIM	**: 17410100180**
Program Studi	**: S1 Sistem Informasi**

FAKULTAS TEKNOLOGI DAN INFORMATIKA
UNIVERSITAS DINAMIKA
2021

Tugas Akhir

ANALISIS SENTIMEN PUBLIK TERHADAP KEBIJAKAN PEMBERLAKUAN PEMBATASAN KEGIATAN MASYARAKATSKALA MIKRO MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE STUDI KASUS TWITTER

Dipersiapkan dan disusun oleh:

Renas Madya Pradhana

NIM: 17410100180

Telah diperiksa, dibahas dan disetujui oleh Dewan Pembahas

Pada: Juli 2021

Susunan Dewan Pembahas

Pembimbing:

I. Dr. M.J. Dewiyani Sunarto
NIDN. 0725076301

II. Julianto Lemantara, S.Kom., M.Eng.
NIDN. 0722108601

Digitally signed by Dewiyani

Digitally signed by Julianto Lemantara
Date: 2021.08.23 09:31:39 +07'00'

Pembahas:

Tutut Wurijanto, M.Kom.
NIDN. 0703056702

TututWurijanto 2021.08.23 15:43:51 +07'00'

Tugas Akhir ini telah diterima sebagai salah satu persyaratan untuk memperoleh gelar Sarjana

Digitally signed by Universitas Dinamika
Date: 2021.08.24 12:17:54 +07'00'

Tri Sagirani, S.Kom., M.MT.

NIDN: 0731017601

Dekan Fakultas Teknologi dan Informatika

UNIVERSITAS DINAMIKA

SURAT PERNYATAAN

PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH

Sebagai mahasiswa Universitas Dinamika, saya:

Nama	: Renas Madya Pradhana
NIM	: 17410100180
Program Studi	: S1 Sistem Informasi
Fakultas	: Fakultas Teknologi dan Informatika
Jenis Karya	: Tugas Akhir
Judul Karya	: **ANALISIS SENTIMEN PUBLIK TERHADAP KEBIJAKAN PEMBERLAKUAN PEMBATASAN KEGIATAN MASYARAKAT SKALA MIKRO MENGGUNAKAN ALGORITMA SUPPORT VECTOR MACHINE STUDI KASUS TWITTER**

Menyatakan dengan sesungguhnya bahwa:

1. Demi pengembangan Ilmu Pengetahuan, Teknologi dan Seni, saya menyetujui memberikan kepada Universitas Dinamika Hak Bebas Royalti Non-Eksklusif (*Non-Exclusive Royalti Free Right)* atas seluruh isi/sebagian karya ilmiah saya tersebut di atas untuk disimpan, dialihmediakan dan dikelola dalam bentuk pangkalan data *(database)* untuk selanjutnya didistribusikan atau dipublikasikan demi kepentingan akademis dengan tetap mencantumkan nama saya sebagai penulis atau pencipta dan sebagai pemilik Hak Cipta.
2. Karya tersebut di atas adalah karya asli saya, bukan plagiat baik sebagian maupun keseluruhan. Kutipan, karya atau pendapat orang lain yang ada dalam karya ilmiah ini adalah semata hanya rujukan yang dicantumkan dalam Daftar Pustaka saya.
3. Apabila dikemudian hari ditemukan dan terbukti terdapat tindakan plagiat pada karya ilmiah ini, maka saya bersedia untuk menerima pencabutan terhadap gelar kesarjanaan yang telah diberikan kepada saya.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

Surabaya, Juli 2021

Menyatakan

Renas Madya Pradhana

NIM. 17410100180

"Eat the frog."

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Dengan ini, penulis persembahkan sebuah hasil karya kecil ini kepada

Orang Tua.

ABSTRAK

Seluruh negara di dunia mengalami masa-masa sulit akibat pandemi *Covid*-19. Krisis muncul di sektor kesehatan dan ekonomi, termasuk Indonesia. Presiden juga meminta agar krisis ini segera ditangani dan harus diselesaikan bersamaan. Berbagai upaya telah dilakukan pemerintah seperti seperti Pemberlakuan Pembatasan Kegiatan Masyarakat (PPKM) skala mikro atau PPKM mikro untuk mencegah penyebaran kasus *corona* (COVID-19). Kebijakan PPKM mikro menimbulkan pro-kontra dari masyarakat karena dampaknya terhadap perekonomian dan tidak efektif menurunkan kasus harian *Covid*-19. Tanggapan dan opini publik disampaikan melalui berbagai media, lebih dari 63,6% dari total pengguna media sosial di Indonesia memiliki akun Twitter. Berdasarkan latar belakang tersebut, penelitian ini berharap dapat menggunakan data dari twitter untuk memahami respon dan persepsi masyarakat Indonesia terhadap kebijakan PPKM mikro dengan membagi respon masyarakat menjadi sentimen positif dan negatif. Algoritma Term Frequency Inverse Document Frequency (TF-IDF) untuk pembobotan. Metode Support Vector Machine untuk mengklasifikasikan kebijakan PPKM mikro. Hasil prediksi tersebut dapat menjadi tolak ukur atau bahan evaluasi bagi pemerintah agar dapat memperpanjang atau tidak kebijakan mengenai pemberlakuan pembatasan kegiatan masyarakat skala mikro sehingga dapat dilakukan penanganan ke evaluasi yang lebih baik. Untuk mengatasi Hasil dari penelitian yang telah dilakukan menunjukkan hasil evaluasi dan validasi menggunakan *k-fold cross validation* pada *support vector machine* menunjukkan rata-rata *cross validation score* sebanyak 96,42%. Hasil pengujian menunjukkan *accuracy* sebanyak 97,13%, hasil *precision* sebanyak 97,58%, dan hasil *recall* sebanyak 99,15%. *Support Vector Machine* (SVM) dapat digunakan untuk melakukan klasifikasi analisis sentimen dengan baik dengan menunjukkan hasil *accuracy* atau hasil sentimen yang benar diprediksi positif dan negatif dari seluruh data *tweet* sangat tinggi. Hasil visualisasi dari diagram pie menunjukkan persentase keseluruhan sentimen pada *tweet* PPKM Mikro sebanyak 12,8% atau 491 cuitan negatif dan 87,2% atau 3574 cuitan positif. Berdasarkan hasil prediksi tersebut dapat menjadi tolak ukur atau bahan evaluasi bagi pemerintah agar dapat memperpanjang kebijakan mengenai pemberlakuan pembatasan kegiatan masyarakat skala mikro.

Kata kunci: *analisis sentimen, ppkm mikro, term frequency-inverse document frequency, support vector machine, twitter.*

KATA PENGANTAR

Puji dan syukur atas kehadirat Tuhan Yang Maha Esa karena atas rahmat dan karunia-Nya, penulis dapat menyelesaikan laporan tugas akhir yang berjudul "Analisis Sentimen Publik terhadap Kebijakan Pemberlakuan Pembatasan Kegiatan Masyarakat skala Mikro Menggunakan Algoritma Support Vector Machine Studi Kasus Twitter" dengan baik dan lancar meskipun penulis sadari bahwa masih ada banyak kekurangan yang ada didalamnya.

Dalam pelaksanaan tugas akhir dan penyelesaian laporan tugas akhir ini, penulis mendapatkan bimbingan dan dukungan dari berbagai pihak. Oleh karena itu, pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Tuhan yang Maha Esa, Allah SWT karena selalu mendengarkan doa yang penulis selalu panjatkan supaya diperlancar untuk pengerjaan tugas akhir ini.
2. Orang tua, Nenek, dan keluarga besar penulis yang selalu memberikan dukungan dan motivasi.
3. Ibu Tri Sagirani, S.Kom., M.MT selaku Dekan Fakultas Teknologi dan Informatika Universitas Dinamika.
4. Bapak Anjik Sukmaaji, S.Kom., M.Eng selaku Ketua Program Studi S1 Sistem Informasi Universitas Dinamika.
5. Ibu Dr. M.J. Dewiyani Sunarto selaku dosen pembimbing yang telah memberikan dukungan penuh berupa motivasi, saran, dan wawasan bagi penulis selama pelaksanaan tugas akhir dan pembuatan laporan tugas akhir.
6. Bapak Julianto Lemantara, S.Kom., M.Eng., OCA., MCTS selaku dosen pembimbing yang telah memberikan dukungan penuh berupa motivasi, saran, dan wawasan bagi penulis selama pelaksanaan tugas akhir dan pembuatan laporan tugas akhir.
7. Bapak Tutut Wurijanto, M.Kom dosen pembahas yang telah menyempurnakan laporan tugas akhir ini.
8. Teman-teman Sistem Informasi yang selalu siap memberikan bantuan, arahan, dan motivasi kepada penulis untuk dapat menyelesaikan laporan tugas akhir ini.
9. *Last but not least, I wanna thank me, I wanna thank me for believing in me, I wanna thank me for doing all this hard work, I wanna thank me for having no days off, I wanna thank me for, for never quitting, I wanna thank me for always*

being a giver, and tryna give more than I receive, I wanna thank me for tryna do more right than wrong, I wanna thank me for just being me at all times.

Semoga Tuhan Yang Maha Esa memberikan rahmat-Nya kepada seluruh pihak yang membantu penulis dalam pelaksanaan tugas akhir dan penyelesaian laporan tugas akhir.

Penulis menyadari di dalam laporan tugas akhir ini masih memiliki banyak kekurangan, meskipun demikian penulis tetap berharap laporan tugas akhir ini dapat bermanfaat bagi semua pihak dan dapat menjadi bahan acuan untuk penelitian selanjutnya.

Surabaya, Juli 2021

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Presiden Joko Widodo mengatakan bahwa dalam satu tahun terakhir, semua negara di dunia mengalami masa-masa sulit akibat pandemi *Covid*-19. Krisis muncul di sektor kesehatan dan ekonomi, termasuk Indonesia. Presiden juga meminta agar krisis ini segera ditangani dan harus diselesaikan bersamaan (Farisa, 2021). Berbagai upaya telah dilakukan pemerintah untuk mencegah penyebaran kasus *Corona* (COVID-19). Mulai dari penerapan Pembatasan Sosial Berskala Besar (PSBB) hingga Penerapan Pembatasan Kegiatan Masyarakat (PPKM) bahkan Pemberlakuan Pembatasan Kegiatan Masyarakat (PPKM) skala mikro atau PPKM mikro. Kebijakan mulai dari PSBB hingga PPKM mikro menimbulkan pro-kontra dari masyarakat karena dampaknya terhadap perekonomian dan tidak efektif menurunkan kasus harian *Covid*-19. Berdasarkan data Satuan Tugas Penanganan *Covid*-19 pada Selasa (26/1/2021) pukul 12.00 WIB, per 2 Maret 2020 kasus pertama diumumkan, Indonesia memiliki 1.012.350 kasus *Covid*-19. (Farisa, 2021). Pada saat yang sama, kebijakan tersebut diyakini berdampak pada penurunan aktivitas ekonomi dan pendapatan masyarakat yang terbatas. Indeks ekonomi saat ini [IKE] pada Januari 2021 adalah 63,0, turun dari 68,6 bulan lalu. Sementara itu, seluruh komponen IKE mengalami penurunan, yang paling parah adalah indeks ketersediaan lapangan kerja sebanyak -10,3 poin menjadi 43,2. Selain itu, dibandingkan dengan enam bulan sebelumnya, kepercayaan konsumen terhadap pendapatan saat ini juga melemah karena penurunan pendapatan reguler seperti upah, remunerasi, upah dan omset (Elena, 2021). Masyarakat yang tinggal di Jawa-Bali juga terbagi. Meski pendapatan menurun, sekitar 44% warga di Bali, Jawa, memilih untuk menerima PPKM Mikro secara ketat. Walaupun risiko tertular *Covid*-19 meningkat, 46% warga tetap memilih untuk menghentikan PPKM Mikro (Makdori, 2021).

Tri Yunis Miko, ahli epidemiologi Universitas Indonesia, menilai pembatasan kegiatan mikro tidak berdampak signifikan terhadap penurunan jumlah kasus aktif corona di Indonesia (Luthfan, 2021). Sependapat dengan ahli epidemiologi,

pengamat kebijakan publik Agus Pambagio menilai penerapan pembatasan kegiatan masyarakat mikro tidak efektif dalam menangani *Covid*-19 (Guritno, 2021). Sedangkan pemerintah mengklaim PPKM mikro efektif menangani kasus positif *Covid*-19 dan angka kematian menurun. Kegiatan PPKM mikro tersebut haruslah mempertimbangkan segala aspek, mulai dari aspek untuk menekan angka kenaikan positif virus *corona* (*Covid*-19), hingga aspek untuk membuat roda perputaran ekonomi seperti pada sektor sektor pertanian, perdagangan, manufaktur, transportasi, dan jasa keuangan tetap berjalan. Semua aspek tersebut haruslah dipertimbangkan secara terperinci agar rencana PPKM mikro dapat berjalan dengan baik dan terhindar dari hal-hal yang justru akan merugikan seperti peningkatan kasus aktif corona sampai aktivitas perekonomian terhenti yang menyebabkan kehilangan pekerjaan dan pendapatan terutama pada masyarakat menengah bawah yang berpenghasilan tidak tetap dan bekerja di sektor informal. Kegiatan PPKM mikro tersebut juga haruslah mempertimbangkan berbagai masukan, diantaranya adalah dengan melihat bagaimana respon dan opini masyarakat terhadap PPKM mikro. Hal ini sangat penting karena pemerintah dapat mempertimbangkan hal tersebut saat menyikapi sikap publik.

Tanggapan dan opini publik disampaikan melalui berbagai media. Berdasarkan data dari layanan manajemen konten *HootSuite* dan agensi pemasaran media sosial *We Are Social* (Kemp, 2021), judul laporannya adalah "Digital 2021", pada tahun 2021 Indonesia memiliki populasi penduduk sebanyak 274,9 juta jiwa. Dengan 202,6 juta jiwa orang Indonesia di antaranya sudah menjadi pengguna internet. Aktivitas internet favorit pengguna internet di Indonesia adalah media sosial, jumlah pengguna media sosial di Indonesia setara dengan 61,8% dari total populasi pada Januari 2021. Jumlah pengguna aktif media sosial di Indonesia yaitu berjumlah 170 juta pengguna. Salah satu media sosial yang paling banyak digunakan oleh masyarakat Indonesia adalah media sosial twitter. Twitter adalah layanan jejaring sosial berupa *microblog* yang memungkinkan penggunanya untuk berbagi pesan berbasis teks yang disebut *tweet*, lebih dari 63,6% dari total pengguna media sosial di Indonesia memiliki akun Twitter (Kemp, 2021). Oleh sebab itu Twitter dapat digunakan secara efektif menjadi sumber penting opini publik karena

berbasis teks untuk bahan penelitian terutama analisis sentimen dibandingkan dengan sumber lain.

Pengumpulan data melalui media sosial diyakini dapat meningkatkan efisiensi di segala aspek. Efisiensi ini mencakup biaya untuk memperoleh data dalam jumlah minimum, dapat memperoleh data secara *real time*, dan menghasilkan data yang berisi informasi yang lebih detail untuk menggambarkan opini publik yang sebenarnya. Dalam banyak penelitian sebelumnya, data twitter juga telah digunakan untuk menganalisis tanggapan dan opini publik. Misalnya penelitian yang dilakukan untuk melihat sentimen analisis terhadap tokoh publik (Pamungkas, 2018), sentimen analisis terhadap bom bunuh diri di surabaya 13 mei 2018 (Ihsan et al., 2019), perbandingan akurasi dan waktu proses algoritma K-NN dan SVM dalam analisis sentimen Twitter (Nasution & Hayaty, 2019), dan penelitian yang dilakukan untuk melihat sentimen analisis publik terhadap Joko Widodo terhadap wabah *Covid*-19 (Hikmawan et al., 2020), perbandingan akurasi metode Naïve Bayes sebanyak 84.58%, Support Vector Machine sebanyak 92.93%, dan k-NN sebanyak 83.70%. Hasil precision dari Naïve Bayes sebanyak 82.14%, SVM sebanyak 95.70% dan k-NN sebanyak 80.66%. Juga hasil recall dari Naïve Bayes sebanyak 85.82%, SVM sebanyak 89.17%, dan k-NN sebanyak 84.13%.

Berdasarkan latar belakang tersebut, penelitian ini berharap dapat menggunakan data dari twitter untuk memahami respon dan persepsi masyarakat Indonesia terhadap kebijakan PPKM mikro, apakah masyarakat lebih banyak beranggapan positif atau negatif sehingga dari hasil prediksi tersebut dapat menjadi tolak ukur atau bahan evaluasi bagi pemerintah agar dapat memperpanjang atau tidak kebijakan mengenai pemberlakuan pembatasan kegiatan masyarakat skala mikro sehingga dapat dilakukan penanganan ke evaluasi yang lebih baik. Untuk mengatasi masalah tersebut, studi ini akan melakukan analisis sentimen dengan membagi respon masyarakat menjadi sentimen positif dan negatif, serta menggunakan metode *Support Vector Machine* untuk mengklasifikasikan kebijakan PPKM mikro, dan menggunakan algoritma *Term Frequency Inverse Document Frequency* (TF-IDF) untuk pembobotan.

1.2 Perumusan Masalah

Berdasarkan permasalahan di atas maka dapat dirumuskan bahwa masalah yang ada yaitu, bagaimana menganalisis tanggapan, dan opini yang terdapat pada Twitter mengenai kebijakan pemberlakuan pembatasan kegiatan masyarakat skala mikro dengan algoritma *Support Vector Machine*.

1.3 Pembatasan Masalah

Berdasarkan rumusan masalah di atas, batasan masalah dalam tugas akhir sebagai berikut:

1. Data analisis sentimen yang digunakan hanya dari Twitter yang menggunakan bahasa Indonesia.
2. Data yang digunakan untuk analisis sentimen berasal dari Twitter dengan kata kunci ppkm mikro.
3. Analisis sentimen menggunakan metode TF-IDF dan *Support Vector Machine*.

1.4 Tujuan Penelitian

Berdasarkan uraian diatas tujuan penelitian yang dibuat yaitu menghasilkan analisis sentimen terkait kebijakan pemberlakuan pembatasan kegiatan masyarakat skala mikro dengan cara mengklasifikasikan sentimen pada data *tweets* di Twitter menggunakan metode *Support Vector Machine* (SVM) sehingga dapat diketahui respon masyarakat apakah lebih banyak beranggapan positif atau negatif terhadap kebijakan pemerintah tersebut.

1.5 Manfaat Penelitian

Berdasarkan analisis sentimen yang dilakukan, diharapkan hasil analisis sentimen dapat digunakan sebagai tolak ukur atau bahan evaluasi terhadap pemerintah mengenai kebijakan PPKM Mikro. Pemerintah dapat mengetahui informasi apa yang tersembunyi dari banyaknya kumpulan data dari Twitter sehingga dapat dilakukan penanganan ke evaluasi yang lebih baik.

BAB II
LANDASAN TEORI

Berikut adalah data dari penelitian terdahulu yang menjadi perbandingan dengan penelitian yang akan dilakukan.

2.1 Penelitian Terdahulu

Tabel 2. 1. Penelitian Terdahulu

Judul	Penulis	Hasil	Akurasi	Pembeda
Analisis Sentimen Terhadap Tokoh Publik Menggunakan Algoritma Support Vector Machine (SVM)	I. Taufik dan S.A. Pamungkas (2018)	Penelitian ini telah berhasil mengimplementasikan Algoritma SVM untuk analisis sentimen pada data *tweet* tentang tokoh publik dengan keyword "Ahok" dan "@teman_ahok". Hal ini diperlihatkan dengan hasil akurasi menggunakan data tweet sebanyak 630 data, dimana 420 data adalah data training dan 210 adalah data testing.	Memiliki tingkat akurasi paling baik sekitar 81%	Penelitian kali ini menggunakan bahasa pemrograman Python, pada data tweet dengan keyword "ppkm mikro", hasil crawling data di Twitter yang dilakukan pada penelitian kali ini didapatkan 4066 data tweets, visualisasi data yang digunakan adalah diagram pie dan word cloud.
Analisis Sentimen Twitter terhadap Bom Bunuh Diri di Surabaya 13 Mei 2018 menggunakan Pendekatan Support Vector Machine	Listaria, Munaffidzul Ihsan, Eky Roza Paradistia, Edy Widodo (2019)	Berdasarkan *tweets* di Twitter sebanyak 104 data tweets berada pada klasifikasi yang benar. Selanjutnya dapat dilihat model machine learning Support Vector Machine mengklasifikasikan kategori sentimen negatif masyarakat sebanyak 1604 tweets dan berada pada klasifikasi yang benar, sehingga tingkat akurasi yang didapatkan adalah 100% atau tidak terdapat kesalahan klasifikasi pada model SVM untuk data penelitian ini.	Hasil klasifikasi sentimen menggunakan pendekatan Support Vector Machine menghasilkan tingkat akurasi sebanyak 100%,	Penelitian kali ini menggunakan bahasa pemrograman Python, pada data tweet dengan keyword "ppkm mikro", hasil crawling data di Twitter yang dilakukan pada penelitian kali ini didapatkan 4066 data tweets, visualisasi data yang digunakan adalah diagram pie.

Tabel 2. 2 Perbandingan Algoritma

Judul	Penulis	Data	Akurasi	Kesimpulan
Perbandingan Akurasi dan Waktu Proses Algoritma K-NN dan SVM dalam Analisis Sentimen Twitter	Muhammad Rangga Aziz Nasution, Mardhiya Hayaty (2019)	Data yang dikumpulkan menggunakan Twitter API yang telah disediakan oleh Twitter. Kata kunci yang dicari adalah tweet yang mengandung "Donald Trump" atau yang berkaitan tentang Donald Trump. Data yang dikumpulkan sebanyak 1113 data. Masing-masing data dipisahkan menjadi data uji dan data latih. Data uji diberi label sedangkan data latih tidak diberi label namun akan diberikan kolom expected untuk perhitungan akurasi dan F1 Score.	Pengujian akurasi dari Analisis Sentimen Terhadap Wacana Politik Pada Media Massa Online Berbahasa Inggris dengan 1. Metode K-Nearest Neighbor dengan akurasi sebanyak 88,1%. 2. Metode Support Vector Machine akurasi sebanyak 88,76%.	Berdasarkan hasil dan pembahasan dari eksperimen yang telah dilakukan, kesimpulan yang bisa ditarik dari penelitian ini adalah: Metode klasifikasi Support Vector Machine memiliki akurasi yang lebih tinggi dibandingkan metode klasifikasi K-Nearest Neighbor dengan validasi K-Fold Cross Validation.
Sentimen Analisis Publik Terhadap Joko Widodo Terhadap Wabah *Covid*-19 Menggunakan Metode Machine Learning	Sisferi Hikmawan, Amsal Pardamean, Siti Nur Khasanah (2020)	Dengan penggunaan dua kata kunci tersebut dapat dihasilkan hasil twit yang fokus hanya pada "Jokowi" dan "*Covid*".	1.Naïve Bayes dengan akurasi 84.58%. 2.Support Vector Machine sebanyak 92.93%. 3.K-Nearest Neighbor dengan akurasi 83.70%.	Berdasarkan pada percobaan-percobaan yang telah dilakukan serta perhitungan akurasi terhadap ketiga metode tersebut, dapat disimpulkan bahwa Support Vector Machine yang terbaik karena memiliki akurasi dan presisi tertinggi. Untuk kedepannya kita perlu menggunakan dataset yang lebih besar dan kompleks lagi serta penyempurnaan preprocessing untuk bahasa Indonesia yang tidak baku.

Berdasarkan penelitian sebelumnya yaitu pada Tabel 1, Tabel 2, Tabel 3 dan Tabel 4, algoritma SVM telah banyak digunakan dalam penelitian, dan hasil

akurasinya lebih baik dibandingkan dengan metode NBC dan k-NN. Oleh karena itu jika dijadikan bahan evaluasi akan diperoleh hasil yang baik. Penelitian ini juga menggunakan proses preprocessing teks dengan metode case cleansing, folding, tokenisasi, filtering, dan stemming. pembobotan menggunakan metode TF-IDF. Dalam penelitian ini, data di Twitter akan mengklasifikasikan opini publik terhadap kebijakan PPKM Mikro sebagai reaksi positif dan negatif. Data yang digunakan dalam penelitian ini adalah data yang diperoleh dari Twitter yang berjumlah sekitar 4066 data.

2.2 Analisis Sentimen

Sentiment analysis merupakan salah satu studi komputasi untuk menggali informasi dan mengklasifikasikan emosi (positif dan negatif) dari pendapat atau pandangan (sentimen) tentang suatu isu atau peristiwa tertentu. Sistem analisis sentimen menggabungkan metode NLP (Natural Language Processing) dan statistik atau machine learning. Pada dasarnya analisis sentimen adalah sebuah klasifikasi, namun pelaksanaannya tidak mudah, karena seperti proses klasifikasi biasa, karena terkait dengan penggunaan bahasa, dimana penggunaan kata-kata ambigu, tidak ada intonasi dalam teks, dan Perkembangan bahasa itu sendiri (Nomleni, 2015). Analisis sentimen sangat penting dalam bidang bisnis seperti melakukan analisa terhadap sebuah produk yang dapat dilakukan secara cepat serta digunakan sebagai alat bantu untuk melihat respon konsumen tentang produk tersebut sehingga dapat sebagai bahan evaluasi yang sangat membantu untuk meningkatkan penjualan produk dan omzet.

2.3 Twitter

Twitter merupakan layanan jejaring sosial dan mikroblog daring yang memiliki konsep yaitu memungkinkan penggunanya untuk mengirim pesan berbasis teks dengan batas 280 karakter kepada pembacanya di seluruh dunia yang bisa digunakan sebagai sarana penyebar informasi kepada semua orang baik yang dikenal maupun tidak, untuk berbagi secara dua arah bagi penggunanya. Twitter adalah situs web yang dimiliki dan dioperasikan oleh Twitter Inc, yang menyediakan jejaring sosial berupa microblog sehingga memungkinkan penggunanya untuk mengirim dan membaca pesan yang disebut tweet. Salah satu

fitur yang terdapat di Twitter adalah Tweet, dan fitur Tweet ini merupakan fitur utama di Twitter. Tweet adalah tweet yang digunakan untuk mempublikasikan dan melihat tweet dari setiap pengguna Twitter. Berdasarkan tweet yang dihasilkan setiap harinya oleh pengguna Twitter, dapat menjadi suatu sumber informasi sehingga dapat dilakukan proses crawling data Twitter dengan menggunakan API

Informasi publik disediakan oleh Twitter. Salah satu Application Program Interface (API) yang digunakan untuk mengakses data Twitter adalah REST API, yang didasarkan pada arsitektur REST saat ini yang digunakan untuk mendesain Web API. Jenis API ini menggunakan *pull strategy* untuk mengambil informasi data dan untuk mengumpulkan informasi, pengguna harus secara eksplisit memintanya (Negara et al., 2016). Akses tweets membutuhkan hak akses berupa *access token, access token secret, consumer key,* dan *consumer secret,*

2.4 Text Mining

Text mining dapat diartikan lebih luas sebagai serangkaian analisis untuk mendapatkan wawasan tentang di mana pengguna berinteraksi dengan koleksi dokumen dari waktu ke waktu. Kumpulan dokumen merupakan sumber data dalam *text mining*. Pola yang menarik tidak ditemukan pada *record database* yang terbentuk, tetapi pada data kata yang tidak terstruktur dalam kumpulan dokumen. (Feldman, dikutip dalam Pristiyanti et al., 2018). Tujuan text mining adalah untuk mendapatkan informasi yang berguna dari sekumpulan dokumen yang besar.

2.5 Text Preprocessing

Proses ini bertujuan untuk membersihkan data yang digunakan dalam proses klasifikasi sentimen dari *noise*. Oleh karena itu, diperlukan suatu proses yang dapat mengubah data yang sebelumnya tidak terstruktur menjadi data terstruktur untuk tahap pemrosesan selanjutnya. Tahapan *preprocessing* dalam penelitian ini meliputi *case folding, filtering, tokenizing, stopword removal,* dan *stemming*. Adapun penjelasan rincinya adalah sebagai berikut :

2.5.1 Cleansing

Cleansing adalah proses menghilangkan simbol yang tidak perlu (seperti tanda baca, angka, dan emoji) dalam dokumen. Pada penelitian ini yang dihilangkan

tidak hanya tanda baca, angka, dan emoji melainkan ada tambahan seperti menghilangkan *url*, *mention*, dan *hashtag* pada dokumen.

Tabel 2. 3 Hasil Tahap *Cleansing*

Sebelum *Cleansing*	Sesudah *Cleansing*
Kemacetan di Jakarta di jam-jam sibuk memang bukan pemandangan baru. Namun di tengah kebijakan PPKM skala mikro, kemacetan tampak cukup parah. #kumparanNEWS https://t.co/XvRhlinROm	Kemacetan di Jakarta di jam jam sibuk memang bukan pemandangan baru Namun di tengah kebijakan PPKM skala mikro kemacetan tampak cukup parah
Jawa Barat (Jabar) menjadi provinsi yang paling sering menerapkan Pemberlakuan Pembatasan Kegiatan Masyarakat (PPKM) Mikro di Indonesia. @PemprovJabar https://t.co/aSNdhojMB8	Jawa Barat Jabar menjadi provinsi yang paling sering menerapkan Pemberlakuan Pembatasan Kegiatan Masyarakat PPKM Mikro di Indonesia

2.5.2 *Case folding*

Case folding merupakan proses penyatuan format huruf kecil dari semua teks dalam dokumen menjadi huruf kecil (lowercase).

Tabel 2. 4 Tahapan Case folding

Sebelum *Case Folding*	Sesudah *Case Folding*
Kemacetan di Jakarta di jam jam sibuk memang bukan pemandangan baru Namun di tengah kebijakan PPKM skala mikro kemacetan tampak cukup parah	kemacetan di jakarta di jam jam sibuk memang bukan pemandangan baru namun di tengah kebijakan ppkm skala mikro kemacetan tampak cukup parah
Jawa Barat Jabar menjadi provinsi yang paling sering menerapkan Pemberlakuan Pembatasan Kegiatan Masyarakat PPKM Mikro di Indonesia	jawa barat jabar menjadi provinsi yang paling sering menerapkan pemberlakuan pembatasan kegiatan masyarakat ppkm mikro di indonesia

2.5.3 Tokenizing

Tokenizing merupakan proses pemisahan atau membagi teks berupa kalimat pada dokumen menjadi *token* atau *term*.

Tabel 2. 5 Tahapan Tokenizing

Sebelum *Tokenizing*	Sesudah *Tokenizing*
kemacetan di jakarta di jam jam sibuk memang bukan pemandangan baru namun di tengah kebijakan ppkm skala mikro kemacetan tampak cukup parah	‘kemacetan’ ‘di’ ‘jakarta’ ‘di’ ‘jam’ ‘jam’ ‘sibuk’ ‘memang’ ‘bukan’ ‘pemandangan’ ‘baru’ ‘namun’ ‘di’ ‘tengah’ ‘kebijakan’ ‘ppkm’ ‘skala’ ‘mikro’ ‘kemacetan’ ‘tampak’ ‘cukup’ ‘parah’
jawa barat jabar menjadi provinsi yang paling sering menerapkan pemberlakuan pembatasan kegiatan masyarakat ppkm mikro di indonesia	‘jawa’ ‘barat’ ‘jabar’ ‘menjadi’ ‘provinsi’ ‘yang’ ‘paling’ ‘sering’ ‘menerapkan’ ‘pemberlakuan’ ‘pembatasan’ ‘kegiatan’ ‘masyarakat’ ‘ppkm’ ‘mikro’ ‘di’ ‘indonesia’

2.5.4 Stopword Removal atau Filtering

Stopword Removal atau *Filtering* adalah proses menghapus kata-kata yang tidak penting dalam deskripsi dengan memeriksa apakah kata-kata *hasil parsing* dalam deskripsi yang termasuk dalam daftar kata penting (*wordlist*) atau tidak penting (*stopword*/*stoplist*). Contoh *Stopwords* untuk Bahasa Indonesia adalah "yang", "dan", "di", "dari" dan lain-lain. Dengan menggunakan daftar Stopwords, maka setiap kata dalam kumpulan akan dicocokkan dengan kata yang sudah ada.

Tabel 2. 6 Tahapan Stopword Removal

Sebelum *Stopword Removal*	Sesudah *Stopword Removal*
'kemacetan' 'di' 'jakarta' 'di' 'jam' 'jam' 'sibuk' 'memang' 'bukan' 'pemandangan' 'baru' 'namun' 'di' 'tengah' 'kebijakan' 'ppkm' 'skala' 'mikro' 'kemacetan' 'tampak' 'cukup' 'parah'	'kemacetan' 'jakarta' 'jam' 'jam' 'sibuk' 'pemandangan' 'tengah' 'kebijakan' 'ppkm' 'skala' 'mikro' 'kemacetan' 'parah'
'jawa' 'barat' 'jabar' 'menjadi' 'provinsi' 'yang' 'paling' 'sering' 'menerapkan' 'pemberlakuan' 'pembatasan' 'kegiatan' 'masyarakat' 'ppkm' 'mikro' 'di' 'indonesia'	'jawa' 'barat' 'jabar' 'provinsi' 'menerapkan' 'pemberlakuan' 'pembatasan' 'kegiatan' 'masyarakat' 'ppkm' 'mikro' 'indonesia'

2.5.5 Stemming

Stemming merupakan suatu proses untuk mengubah bentuk kata menjadi akar kata dasar sesuai dengan struktur morfologi bahasa Indonesia yang baik dan benar. *Stemming* umumnya digunakan dalam teks bahasa Inggris, karena teks bahasa Inggris memiliki struktur imbuhan tetap dan mudah diolah, sedangkan *Stemming* bahasa Indonesia memiliki struktur batang yang rumit / kompleks, sehingga pemrosesannya agak sulit. Contoh Algoritma *Stemming* berbahasa Indonesia yaitu Algoritma Porter dan Algoritma Nazief & Adriani.

Berdasarkan hasil penelitian yang dilakukan (Agusta, 2009), kesimpulan dari perbandingan antara Algoritma Porter dengan Algoritma Nazief & Adriani, adalah algoritma Porter membutuhkan waktu yang lebih singkat dibandingkan dengan algoritma Nazief dan Adriani untuk stemming, namun dibandingkan dengan algoritma Nazief dan Adriani untuk stemming, akurasi stemming menggunakan algoritma Porter lebih rendah. Jadi, *Stemming* yang digunakan dalam penelitian ini adalah *Stemming* algoritma Nazief dan Adriani.

Tabel 2. 7 Tahapan Stemming

Sebelum *Stemming*	Sesudah *Stemming*
'kemacetan' 'jakarta' 'jam' 'jam' 'sibuk' 'pemandangan' 'tengah' 'kebijakan' 'ppkm' 'skala' 'mikro' 'kemacetan' 'parah'	'macet' 'jakarta' 'jam' 'jam' 'sibuk' 'pandang' 'tengah' 'bijak' 'ppkm' 'skala' 'mikro' 'macet' 'parah'
'jawa' 'barat' 'jabar' 'provinsi' 'menerapkan' 'pemberlakuan' 'pembatasan' 'kegiatan' 'masyarakat' 'ppkm' 'mikro' 'indonesia'	'jawa' 'barat' 'jabar' 'provinsi' 'terap' 'laku' 'batas' 'giat' 'masyarakat' 'ppkm' 'mikro' 'indonesia'

2.6 Term Frequency-Inverse Document Frequency (TF-IDF)

Metode TF-IDF menurut Abdul (dikutip dalam Amrizal, 2018) merupakan metode untuk menghitung bobot setiap kata yang paling umum digunakan pada *information retrieval*. Metode ini juga terkenal efisien, mudah dan memiliki hasil yang akurat. Metode TF-IDF merupakan salah satu metode pembobotan term yang digunakan sebagai metode perbandingan untuk metode pembobotan baru. Dalam metode ini, bobot istilah *term* t dalam dokumen dapat dihitung dengan mengalikan *Term Frequency* dengan *Inverse Document Frequency*. TF-IDF adalah ukuran statistik yang digunakan untuk mengevaluasi pentingnya sebuah kata dalam sebuah dokumen atau sekelompok kata. Untuk satu dokumen, setiap kalimat diperlakukan sebagai dokumen. Frekuensi kata dalam dokumen tertentu menunjukkan pentingnya kata dalam dokumen. Frekuensi dokumen yang berisi kata tersebut menunjukkan seberapa umum kata tersebut. Jika kata lebih sering muncul dalam satu dokumen, bobot kata lebih besar, dan jika muncul di banyak dokumen, bobotnya lebih kecil (Agung dalam (Amrizal, 2018)). Perhitungan *term frequency* dijabarkan dengan rumus:

$$TF = ft, d \; / \sum_{t' \epsilon d} ft', d \qquad (1)$$

sebagai hasil pembagian antara jumlah kemunculan kata pada suatu dokumen dengan total jumlah kata yang terkandung pada dokumen tersebut. Perhitungan *inverse document frequency* yang dijabarkan dengan rumus: $IDF(t) = \log \frac{1+n}{1+df(t)} + 1 \qquad (2)$

sebagai hasil log pembagian antara n = total jumlah dokumen pada suatu *corpus* dengan jumlah dokumen yang mengandung term tertentu. Setelah ditemukan hasil perhitungan dari *term frequency* dan *inverse document frequency* maka dilakukan perhitungan pembobotan sebagai hasil perkalian antara hasil perhitungan dari *term frequency* dengan *inverse document frequency.* Algoritma

TF-IDF digunakan rumus untuk menghitung bobot (W) tiap dokumen terhadap kata kunci dengan rumus yaitu:

$$Wdt = tfdt * Idft \quad (3)$$

Dimana:

df = banyak dokumen yang mengandung kata yang dicari.

Wdt = bobot dokumen ke- d terhadap kata ke- t

$tfdt$ = banyaknya kata yang dicari pada sebuah dokumen

$Idft$ = Inversed Document Frequency ($log\ (N/df)$)

N = total dokumen

Normalisasi memungkinkan untuk mempertimbangkan hubungan relatif antara frekuensi kata sambil menghilangkan pengaruh jumlah kata total.Perhitungan *L2 normalization* yang dijabarkan dengan rumus:

$$\frac{v}{||v||_2} = \frac{v}{\sqrt{(v_1^2+v_2^2+\cdots+v_n^2}} \quad (4)$$

v/||v||2 = v/√((v1^2+v2^2+⋯+vn^2) sebagai hasil pembagian antara bobot kata dari tf idf dengan total jumlah bobot kata yang dipangkatkan lalu diakarkan dari dokumen tersebut.

2.7 Support Vector Machine

Support Vector Machine (SVM) Metode klasifikasi yang digunakan Pembelajaran mesin (*supervised learning*) adalah memprediksi kategori berdasarkan model atau pola hasil dari proses training. Pada permasalahan dalam pengklasifikasian SVM *non-linear* ada beberapa fungsi *kernel* yang umum digunakan yaitu:

a. *Kernel Linear*

$$K\left(x,y\right) = x.y \quad (5)$$

Menggunakan fungsi *kernel* merupakan salah satu cara untuk mendapatkan hasil klasifikasi terbaik. Untuk klasifikasi temukan hyperplane atau bidang pemisah (*Decision boundary*) dengan kelas lain, dalam hal ini bidang pemisah Berperan dalam memisahkan *tweet* sentimen positif (Ditandai sebagai +1) *tweet* negatif (Ditandai sebagai -1). *Hyperplane* yang baik bisa didapatkan dengan

memaksimalkan jarak *margin.* W juga memiliki beberapa fitur (w_i). Formulasi yang digunakan adalah meminimalkan nilai margin:

$$\frac{1}{2}||w||^2 = \frac{1}{2}({w_1}^2+{w_2}^2+{w_3}^2 \ldots + {w_6}^2) \quad (6)$$

Berikut, diasumsikan jika pada kelas -1 dan +1 dapat terpisah secara sempurna oleh h*yperplane*, yang dimana dapat diberikan sebuah definisi dengan menggunakan syarat sebagai berikut :

$$y_i(w_i.x_i + b) \geq 1, i = 1,2,3, \ldots\ldots, N \quad (7)$$

Jika w_i berada di kelas +1, untuk kelas positif maka dapat dituliskan sebuah definisi berikut ini :

$$y_i(w_1.x_1 + w_2.x_2 + w_3.x_3 \ldots + w_6.x_6 + b) \geq +1 \quad (8)$$

Sedangkan, Jika w_i berada di kelas -1, untuk kelas negatif maka dapat dituliskan sebuah definisi berikut ini :

$$y_i(w_1.x_1 + w_2.x_2 + w_3.x_3 \ldots + w_6.x_6 + b) \geq -1 \quad (9)$$

sehingga didapatkan beberapa persamaan sesuai dengan data. Berdasarkan persamaan, maka didapatkan nilai dari setiap *corpus*. Untuk mendapatkan nilai dari masing-masing atribut, dilakukan perhitungan linear dengan melakukan eliminasi pada masing-masing persamaan. Perhitungan eliminasi w6 (dengan cara perkalian silang indeks w6). Perhitungan nilai w_i dengan menggunakan persamaan, dengan mensubstitusi w_i sebelumnya. Berdasarkan perhitungan diatas, dapat diketahui margin atau $w_i \; dan \; b$. Dengan syarat:

$$y = \begin{cases} +1 \; jika \; w \; . \; z + b > 0 \\ -1 \; jika \; w \; . \; z + b < 0 \end{cases} \quad (10)$$

2.8 K-fold Cross Validation

Teknik validasi yang digunakan dalam penelitian ini adalah *k-fold cross-validation*. Pembagian data dilakukan sehingga ukuran setiap *subset (fold)* sama. Proses *k-fold cross-validation* menghasilkan nilai yang disebut *performance value*. Evaluasi performansi adalah menguji hasil klasifikasi dengan mengukur nilai kinerja dari sistem yang dibuat. Evaluasi di atas menggunakan parameter uji berupa akurasi yang perhitungannya diperoleh dari tabel *coincidence* (Anjasmoros, 2020).

Tabel 2. 8 Confusion Matrix

Dokumen	Kelas Aktual

	Negative	*Positive*
Kelas Prediksi	*True Negative (TN)*	*False Positive (FP)*
	False Negative (FN)	*True Positive (TP)*

Keterangan :

a. True Positive (TP) = jumlah dokumen aktual yang *positive* dan diprediksi *positive*.
b. False Positive (FP) = jumlah dokumen aktual yang *negative* dan diprediksi *positive*.
c. False Negative (FN) = jumlah dokumen aktual yang *positive* dan diprediksi *negative*.
d. True Negative (TN) = Jumlah dokumen aktual yang *negative* dan diprediksi *negative*.

Berdasarkan *matrix confusion*, maka dapat dihitung nilai *accuracy*, *precision*, dan *recall*. *Accuracy* menggambarkan seberapa akurat model dalam mengklasifikasikan dengan benar dengan rumus sebagai berikut:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (11)$$

Precision menggambarkan akurasi antara data yang diminta dengan hasil prediksi yang diberikan oleh model. dengan rumus sebagai berikut:

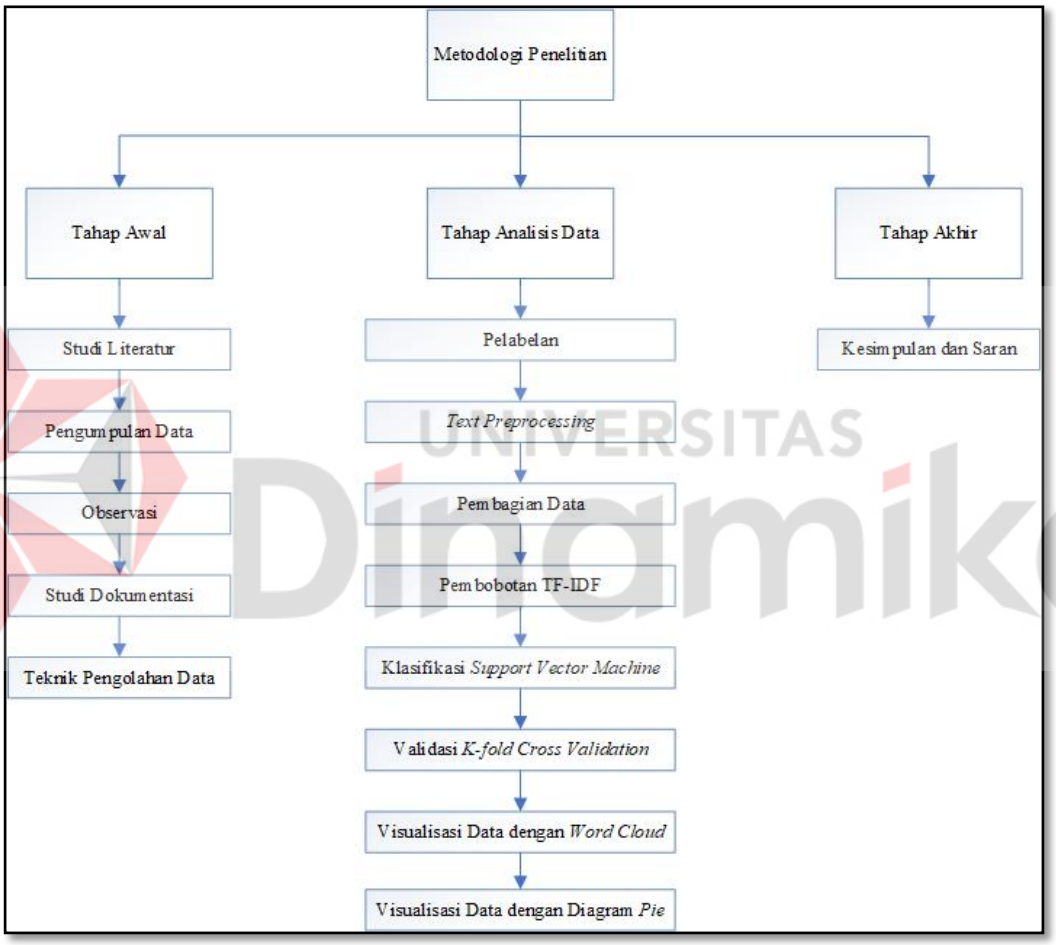
$$Precision = \frac{TP}{TP+FP} \quad (12)$$

Recall menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi.dengan rumus sebagai berikut:

$$Recall = \frac{TP}{TP+FN} \quad (13)$$

BAB III
METODE PENELITIAN

Dalam penelitian ini untuk melakukan analisis sentimen public terhadap kebijakan pemberlakuan pembatasan kegiatan masyarakat skala mikro menggunakan algoritma *support vector machine* pada twitter akan melalui tiga tahapan yaitu tahap awal, tahap analisis data, dan tahap akhir.



Gambar 3. 1 Diagram Metode Penelitian

3.1 Tahap Awal

Dalam Tahap awal, ada dua tahapan penelitian yang dilakukan dalam penelitian ini untuk melakukan analisis sentimen yaitu studi literatur dan pengumpulan data. Pengumpulan data sendiri dibagi menjadi tiga yaitu observasi, studi dokumentasi dan teknik pengolahan data.

3.1.1 Studi Literatur

Penulis menggunakan studi literatur untuk menghimpun data-data atau jurnal-jurnal yang berhubungan dengan topik yang diangkat dalam analisis sentimen. Dengan memahami studi kasus dan metode pada jurnal yang sudah ada ataupun yang serupa dengan masalah yang terjadi pada Twitter. Sebagai bahan perbandingan untuk melakukan analisis sentimen pada penelitian ini.

3.1.2 Pengumpulan Data

Pada tahap pengumpulan data, ada tiga tahapan penelitian yang dilakukan dalam untuk melakukan teknik pengumpulan data yaitu observasi, studi dokumentasi dan teknik pengolahan data. Pengumpulan data sendiri dibagi menjadi tiga yaitu observasi, studi dokumentasi dan teknik pengolahan data. Pada tahap ini penulis melakukan pengumpulan data yang diambil dari Twitter yang memuat *tweets*.

3.1.3 Observasi

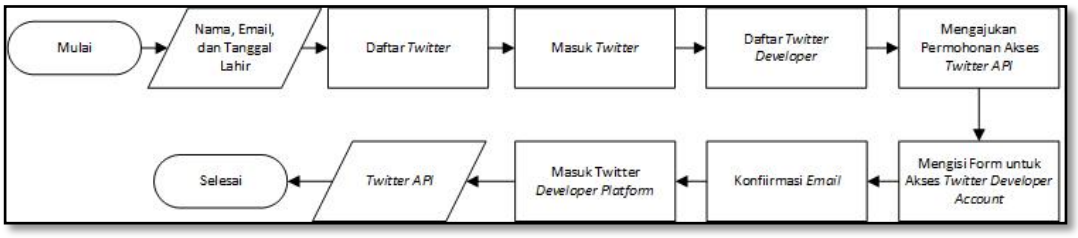
Observasi dalam penelitian ini adalah observasi yang dilakukan dengan mengamati dan memahami objek yang akan diteliti, yaitu dengan mengamati cuitan/*tweet dengan* kata kunci ppkm mikro yang memuat berbagai macam cuitan yang membuat penulis melakukan penelitian terkait dengan cuitan yang ada pada Twitter.

3.1.4 Studi Dokumentasi

Studi dokumentasi dilakukan dengan mendapatkan dan mengumpulkan data dari cuitan dengan kata kunci ppkm mikro di Twitter yang dapat dilihat Gambar L1. 1. Hasil Studi Dokumentasi pada Lampiran 1 Metode Penelitian.

3.1.5 Teknik Pengolahan Data

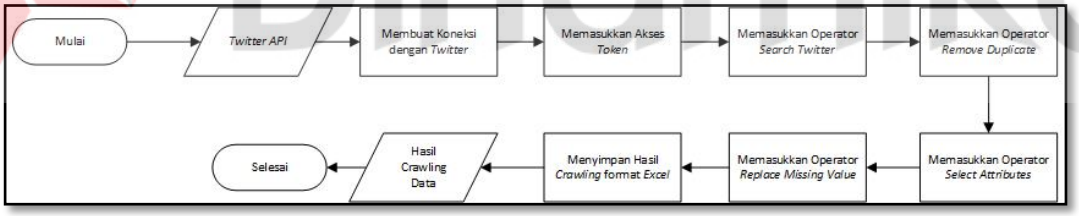
Teknik pengolahan data yang digunakan adalah teknik *crawling* data menggunakan RapidMiner. Data awal yang dibutuhkan pada penelitian ini adalah API Twitter, dengan melakukan *login* ke Twitter API Authorization. Untuk mengakses API Twitter memerlukan akun Twitter, aplikasi, dan token akses yang dapat dilihat pada Gambar 3. 2. Diagram Alir Mendapatkan Twitter API.



Gambar 3. 2 Diagram Alir Mendapatkan Twitter API

Langkah pertama untuk mendapatkan *Twitter API* adalah mendaftar akun menggunakan nama, email dan tanggal lahir pada twitter dan masuk ke dalam Twitter menggunakan akun tersebut. Setelah dapat masuk ke dalam Twitter maka buka *website* Twitter *Developer* untuk melakukan pendaftaran dan ajukan permohonan akses Twitter *API* dengan mengisi form untuk akses Twitter *Developer Account*. Lakukan konfirmasi email dan masuk ke *Twitter Developer Platform*, maka Twitter *API* berhasil didapatkan.

Akses *API* Twitter tersebut digunakan untuk mengambil data-data atau cuitan Twitter dengan kata kunci ppkm mikro pada tanggal 7 April 2021 hingga 13 April 2021. Jumlah *crawling* data dari Twitter mengenai cuitan ppkm mikro yang dengan menggunakan Bahasa Indonesia sebanyak 4066 cuitan yang dapat dilihat pada Gambar 3. 3. Diagram Alir Mengambil Data Twitter.



Gambar 3. 3. Diagram Alir Mengambil Data Twitter

Langkah pertama untuk mengambil data *tweet* pada Twitter menggunakan RapidMiner adalah membuat koneksi dengan *connection type* yang dipilih yaitu Twitter, lalu *request access token* dengan membuka *url* yang telah disediakan dan pilih *authorize app* agar memberikan otorisasi *RapidMiner Social Media Extension* untuk mengakses akun. Setelah mendapat otorisasi maka akan mendapatkan *code* 6 digit, *copy code* tersebut ke dalam kolom yang telah disediakan di dalam *RapidMiner*, Maka *access token* otomatis terisi. Setelah koneksi dapat tersambung maka langkah selanjutnya adalah memasukkan operator *search* twitter ke dalam *process*, pada operator *search* twitter terdapat beberapa parameter yang harus diisi,

untuk parameter *connection source* pilih *repository*, parameter *connection entry* pilih koneksi yang sudah dibuat, parameter *query* masukkan "ppkm mikro", parameter *result type* pilih *recent or popular*, parameter *limit* atur 9999, parameter *language* masukkan "id", dan untuk parameter *until* pilih 13 April 2021. Berikut hasil *crawling* data rapidminer yang ditampilkan Gambar L1. 2. Hasil Crawling Data RapidMiner pada Lampiran 1. Metode Penelitian. setelah mengambil data maka langkah selanjutnya adalah memasukkan operator *remove duplicate* untuk menghapus data yang sama, pada *remove duplicate* untuk *attribute filter type* pilih *single* dan *attribute* pilih *Text*. Lalu memasukkan operator *select attributes* untuk mengambil *text* (*tweet)* saja, pada operator *select attributes* untuk parameter *attribute filter type* pilih *single* dan *attribute* pilih *Text*. Selanjutnya memasukkan operator *replace missing value* untuk mengisi data yang cacat seperti tidak diketahui nilainya atau data yang hilang dengan nilai rata–rata yang sering muncul dan juga menghapus atributnya, pada operator *replace missing value* untuk *attribute filter type* pilih *single*, pada *attribute* pilih *Text*, dan untuk *default* pilih *average*. Langkah terakhir untuk pengambilan data adalah memasukkan operator *write Excel* ke dalam Process, Parameter *excel file* merupakan lokasi penyimpanan file hasil dari crawling data yang akan disimpan dan untuk parameter *file format* pilih *xlsx*. Berikut tangkapan layar seluruh proses *crawling* data pada rapidminer yang dapat dilihat Gambar L1. 3. Crawling Data RapidMiner pada Lampiran 1. Metode Penelitian.

3.2 Tahap Analisis Data

Dalam Tahap analisis data, *software* yang digunakan dalam penelitian ini adalah Microsoft Excel 2016 dan python dengan menggunakan Google Collaboratory. Microsoft Excel 2016 digunakan untuk mengolah hasil *crawling* data twitter untuk pelabelan. Setelah dilakukan pelabelan maka melakukan tahap *text preprocessing*, Setelah melewati proses *text preprocessing*, maka hasil *crawling* data dibagi menjadi data *training* dan data *testing*. Data *training* yang telah dilabeli positif dan negative akan digunakan untuk melatih algoritma, sedangkan data *testing* akan digunakan untuk menguji kinerja algoritma yang dilatih sebelumnya saat menentukan data baru yang belum pernah terlihat sebelumnya. Pada penelitian ini menggunakan data *training* sebanyak 70% yaitu

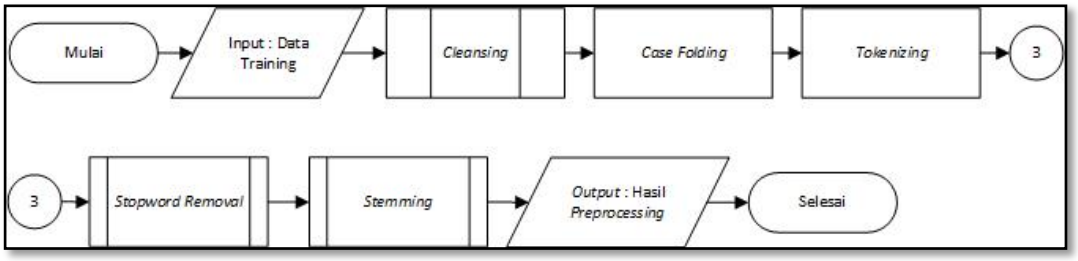
sebanyak 2846 cuitan dan data *testing* sebanyak 30% sebanyak 1.220 cuitan dari total data *tweet* yaitu 4066 cuitan. Lalu dilakukan pembobotan *term frequency-inverse document frequency,* klasifikasi menggunakan algoritma *support vector machine,* validasi *k-fold cross validation*, visualisasi data dengan *word cloud,* dan visualisasi data dengan diagram pie. *Python* digunakan untuk lima tahapan penelitian yang dilakukan dalam penelitian ini mulai dari *text preprocessing*, *term frequency-inverse document frequency, support vector machine*, *k-fold cross validation*, *word cloud,* dan diagram *pie.*

3.2.1 Pelabelan

Hasil crawling data berbentuk file excel yang didapatkan dari aplikasi rapidminer, pelabelan dilakukan dengan melabeli data secara manual dengan bantuan 2 sukarelawan. Pada tahapan ini dilakukan pemberian label negatif dan angka label positif pada label data training. Pembagian data dan pelabelan ini dikerjakan dengan perangkat pembantu yaitu Microsoft Excel 2016. Berikut adalah contoh kamus positif negatif yang ditampilkan Tabel L1. 1. Contoh Kamus Positif dan Negatif pada Lampiran 1. Metode Penelitian.

3.2.2 Text Preprocessing

Data Training tersebut akan dimasukkan pada tahap *Text preprocessing*. Diagram Alir *Preprocessing* dapat dilihat pada Gambar 3. 4. Diagram Alir *Text Preprocessing*.



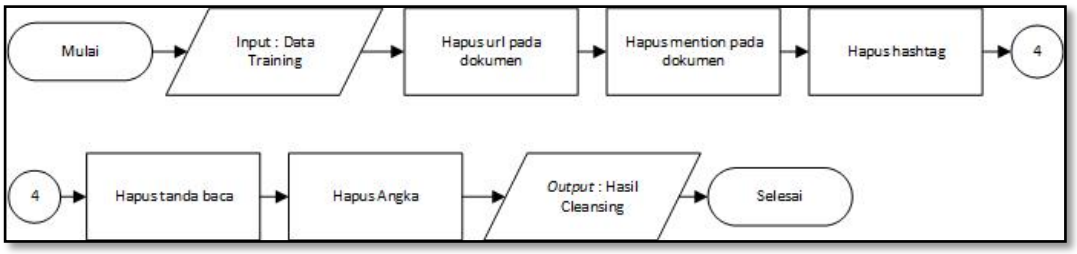
Gambar 3. 4. Diagram Alir *Text Preprocessing*

A. *Input* data

Proses *input* data menggunakan *library pandas* untuk memasukkan atau membaca data dengan format *excel* ke dalam python.

B. *Cleansing*

Proses *cleansing* menggunakan *library re* untuk menghilangkan komponen tertentu pada cuitan seperti tanda baca, angka, *url*, *mention*, dan *hashtag*. Diagram Alir *Cleansing* dapat dilihat pada Gambar 3. 5. Diagram Alir *Cleansing.*



Gambar 3. 5. Diagram Alir *Cleansing*

C. *Case folding*

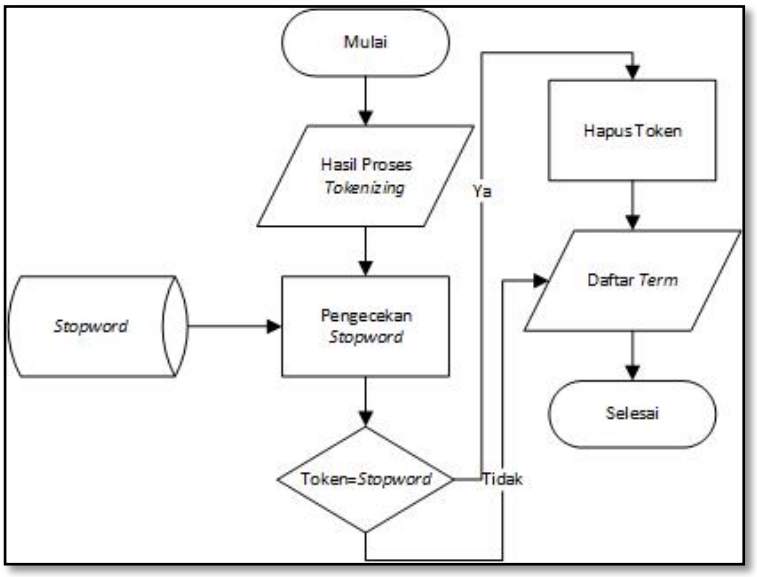
Proses *cleansing* menggunakan *library re* dan *function remove punctuation* untuk penyatuan format huruf kecil dari semua teks dalam dokumen menjadi huruf kecil (*lowercas*e).

D. *Tokenizing*

Proses t*okenizing* menggunakan *library nltk* untuk pemisahan atau membagi teks berupa kalimat pada dokumen menjadi *token* atau *term*.

E. *Stopword Removal* atau *Filtering*

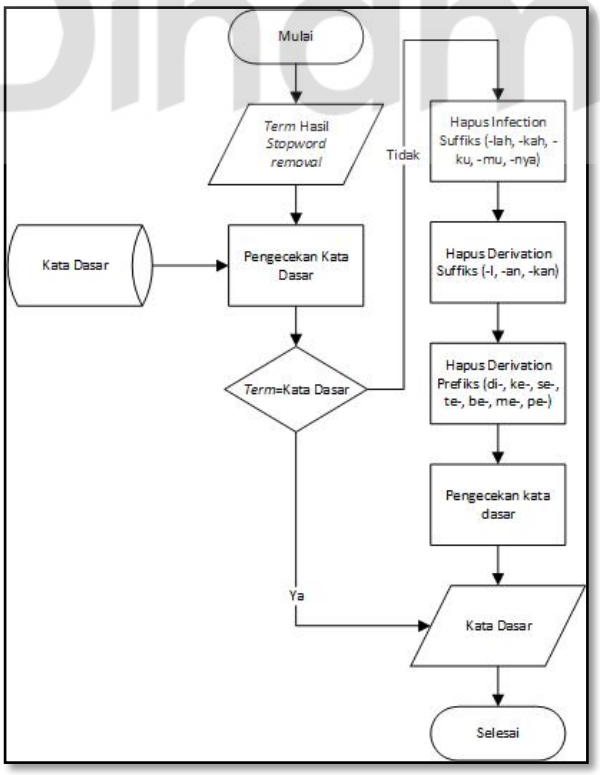
Proses *stopword removal* atau *filtering* menggunakan *library sastrawi* untuk menghapus kata-kata sesuai dengan kata-kata yang terdapat dalam *stopword* atau *stoplist*. Diagram Alir *Stopword Removal atau Filtering* dapat dilihat pada Gambar 3. 6. Diagram Alir *Stopword Removal.*



Gambar 3. 6. Diagram Alir *Stopword Removal*

F. Stemming

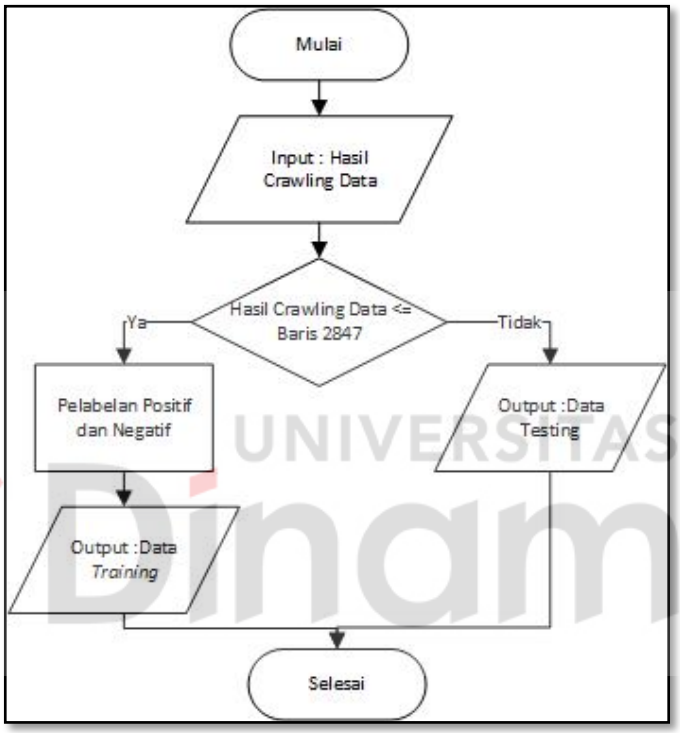
Proses *stemming* menggunakan *library sastrawi* untuk mengubah kata menjadi bentuk kata dasar. Diagram alir Stemming dapat dilihat pada Gambar 3. 7. Diagram Alir *Stemming*.



Gambar 3. 7. Diagram Alir *Stemming*

3.2.3 Pembagian Data

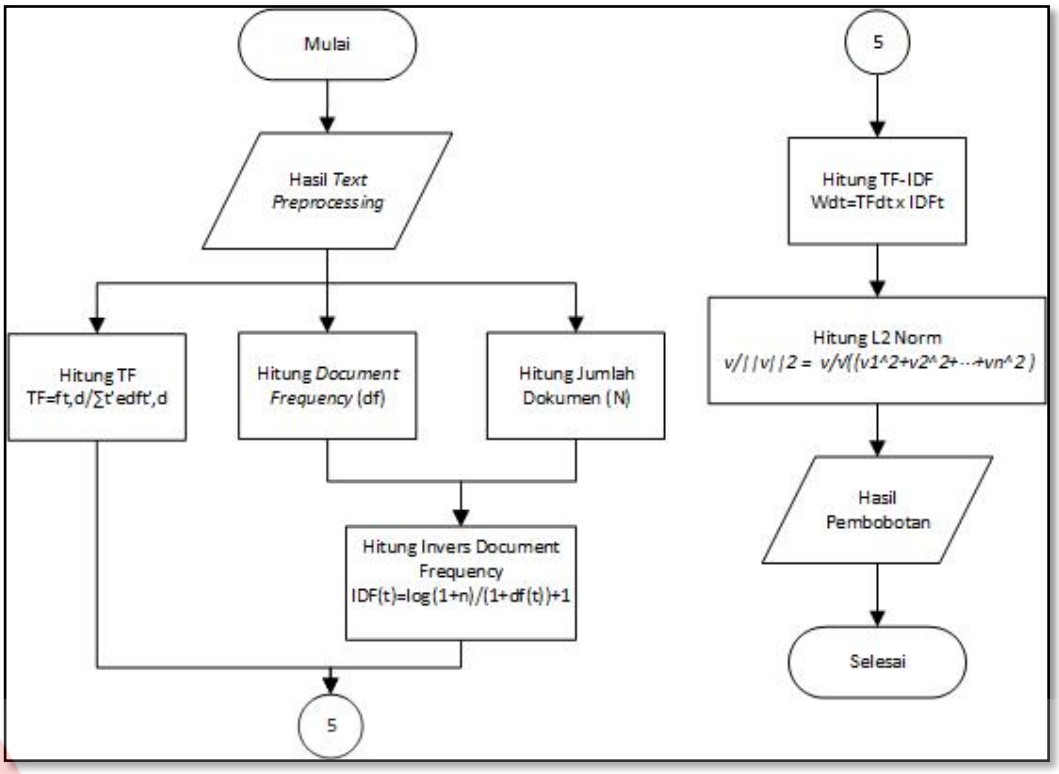
Hasil *crawling* data twitter dilakukan pembagian menjadi data *training* sebanyak 70% dan data *testing* sebanyak 30%. Dengan jumlah data *training* sebanyak 2846 cuitan data *testing* sebanyak 1.220 cuitan. Semakin banyak data *training* maka semakin baik model klasifikasi. Semakin banyak data *testing* semakin perkiraan kesalahan lebih akurat. Diagram alir pembagian data dapat dilihat pada Gambar 3. 8. Diagram Alir Pembagian Data.



Gambar 3. 8. Diagram Alir Pembagian Data

3.2.4 Pembobotan Term Frequency-Inverse Document Frequency

Pembobotan TF-IDF menggunakan *library TfidfVectorizer* untuk mengubah data yang berupa kata menjadi data numerik dan memiliki bobot, juga untuk menyaring data yang akan diproses. Dibawah ini adalah Diagram Alir Klasifikasi *Term Frequency-Inverse Document Frequency* (TF-IDF) yang dapat dilihat pada Gambar 3. 9. Diagram Alir TF-IDF.



Gambar 3. 9. Diagram Alir TF-IDF

Berikut adalah data training yang belum memasuki proses *text preprocessing* yang dapat dilihat Pada Tabel L6. 1. pada Lampiran 6. Bukti Perhitungan Manual TF-IDF. Berikut adalah data training yang telah disesuaikan dengan python yang telah melewati proses *text preprocessing* dan akan digunakan untuk ilustrasi perhitungan *term frequency-inverse document frequency* yang ditampilkan pada Tabel L6. 2 pada Lampiran 6. Bukti Perhitungan Manual TF-IDF. Pada tabel Tabel L6. 3. pada Lampiran 6. Bukti Perhitungan Manual TF-IDF dijelaskan bahwa data training pada tabel di atas akan dilakukan perhitungan *term frequency* yang dijabarkan dengan rumus: $TF = ft, d \; / \sum_{t' \epsilon d} ft', d$ sebagai hasil pembagian antara jumlah kemunculan kata pada suatu dokumen dengan total jumlah kata yang terkandung pada dokumen tersebut. Berikut adalah hasil perhitungan *term frequency* yang ditampilkan Tabel L6. 3. pada Lampiran 6. Bukti Perhitungan Manual TF-IDF

Setelah menemukan hasil dari *term frequency* maka langkah selanjutnya adalah dilakukan perhitungan *inverse document frequency* yang dijabarkan dengan rumus: $IDF(t) = \log \frac{1+n}{1+df(t)} + 1$ sebagai hasil log pembagian antara n = total

jumlah dokumen pada suatu *corpus* dengan jumlah dokumen yang mengandung term tertentu. Berikut adalah hasil perhitungan *inverse document frequency* yang ditampilkan Tabel L6. 4. pada Lampiran 6. Bukti Perhitungan Manual TF-IDF Setelah ditemukan hasil perhitungan dari *term frequency* dan *inverse document frequency* maka dilakukan perhitungan pembobotan dengan rumus: $Wdt = tfdt * Idft$ sebagai hasil perkalian antara hasil perhitungan dari *term frequency* dengan *inverse document frequency.* Hasil pembobotan TF-IDF ditampilkan Tabel L6. 5. pada Lampiran 6. Bukti Perhitungan Manual TF-IDF

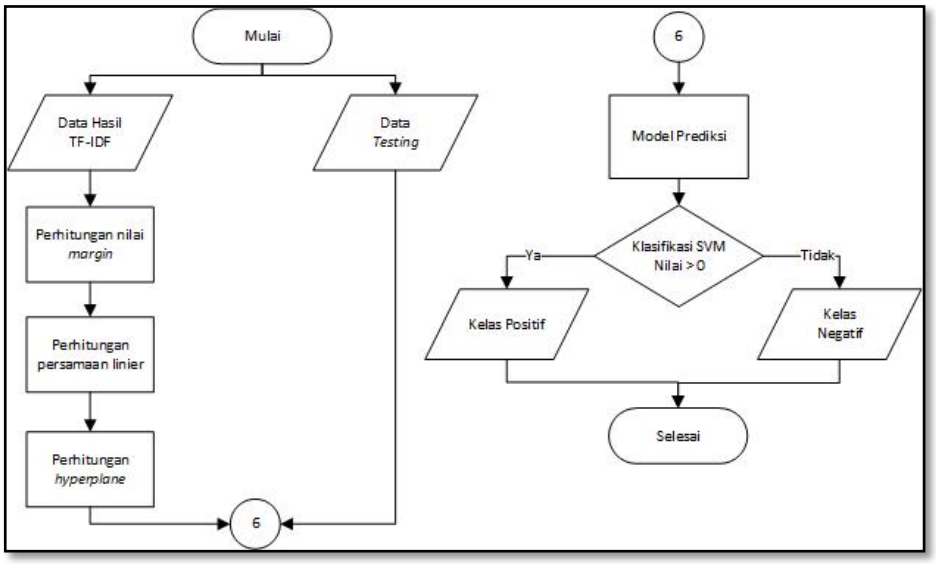
Normalisasi memungkinkan untuk mempertimbangkan hubungan relatif antara frekuensi kata sambil menghilangkan pengaruh jumlah kata total. Perhitungan *L2 normalization* yang dijabarkan dengan rumus: $\frac{v}{||v||_2} = \frac{v}{\sqrt{(v_1^2+v_2^2+\cdots+v_n^2}}$ sebagai hasil pembagian antara bobot kata dari tf idf dengan total jumlah bobot kata yang dipangkatkan lalu diakarkan dari dokumen tersebut. Berikut adalah hasil perhitungan *L2 normalization* yang ditampilkan pada Tabel 3. 2. Hasil L2 Normalization.

Tabel 3. 1. Hasil L2 Normalization

Token	L2 NORM						
	D1	D2	D3	D4	D5	D6	D7
covid	0,452	0,283	0,577	0	0,283	0,242	0,219
efektif	0	0,780	0	0,698	0,390	0	0,302
mikro	0,506	0,283	0,577	0,506	0,283	0,242	0,219
positif	0	0	0	0	0	0,908	0,821
ppkm	0,506	0,283	0,577	0,506	0,283	0,242	0,219
turun	0,698	0,390	0	0	0,780	0	0,302

3.2.5 Klasifikasi Support Vector Machine

Proses klasifikasi *support vector machine* menggunakan *library scikit-learn.* Diagram Alir Klasifikasi *Support Vector Machine* dapa dilihat pada Gambar 3. 10. Diagram Alir Klasifikasi *Support Vector Machine*.



Gambar 3. 10. Diagram Alir Klasifikasi *Support Vector Machine*

Berdasarkan hasil perhitungan pada Lampiran 5. Bukti Perhitungan Manual TF-IDF, maka data numerik dapat digunakan pada tahap berikutnya yaitu pengklasifikasian data menggunakan algoritma SVM. Berikut adalah data training yang telah disesuaikan dengan data yang ada di python dan telah dilakukan pembobotan yang akan digunakan proses perhitungan klasifikasi menggunakan algoritma SVM ditampilkan Tabel L5. 1. pada Lampiran 5. Bukti Perhitungan Manual SVM.

Karena ada 6 fitur (covid, efektif, mikro, positif, ppkm, turun), maka W juga akan memiliki 6 fitur (w1, w2, w3, w4, w5, w6). Formulasi yang digunakan adalah meminimalkan nilai margin:

$$\frac{1}{2}||w||^2 = \frac{1}{2}({w_1}^2+{w_2}^2+{w_3}^2 \ldots + {w_6}^2)$$

Dengan menggunakan syarat sebagai berikut :

$y_i(w_i.x_i + b) \geq 1, i = 1,2,3, \ldots\ldots, N$

$y_i(w_1.x_1 + w_2.x_2 + w_3.x_3 \ldots + w_6.x_6 + b) \geq +1$ untuk kelas positif

$y_i(w_1.x_1 + w_2.x_2 + w_3.x_3 \ldots + w_6.x_6 + b) \geq -1$ untuk kelas negatif

sehingga didapatkan beberapa persamaan yang dapat dilihat pada Lampiran 9. Bukti Perhitungan Manual SVM. Berdasarkan tersebut, maka didapatkan nilai dari setiap *corpus*. Untuk mendapatkan nilai dari masing-masing atribut, lakukan perhitungan linear dengan melakukan eliminasi pada masing-masing persamaan dapat dilihat pada Lampiran 5. Bukti Perhitungan Manual SVM, dapat diketahui

w1 = 0,004; w2 = 0,913; w3 = 0,640; w4 = -3,181; w5 = 0,001; w6 = 0,537; b = 0,521. Dengan syarat:

$$y = \begin{cases} +1 \; jika \; w \; . \; z + b > 0 \\ -1 \; jika \; w \; . \; z + b > 0 \end{cases}$$

Maka data yang bernilai di atas 0 adalah data dalam kelas positif. Sedangkan data yang bernilai di bawah 0 adalah data dalam kelas negatif. Setelah itu data *testing* yang dapat dilihat tabel L5. 2 pada Lampiran 5. Bukti Perhitungan Manual SVM akan dihitung terlebih dahulu bobotnya menggunakan *term frequency-inverse document frequency* yang dapat dilihat Tabel L5. 3. pada Lampiran 5. Bukti Perhitungan Manual SVM. Setelah ditemukan bobot dari setiap kata yang terdapat pada data *testing,* maka dapat diuji menggunakan dengan menggunakan margin yang telah ditemukan. Untuk perhitungannya dapat dilihat pada Lampiran 5. Bukti Perhitungan Manual SVM.

Berdasarkan hasil hitung Lampiran 5. Bukti Perhitungan Manual SVM, dokumen 8 terprediksi secara benar dikarenakan dokumen dilabeli positif dan hasil prediksi positif dengan hasil perhitungannya sebesar 1,767. Dokumen 9 terprediksi secara benar dikarenakan dokumen dilabeli negatif dan hasil prediksi negatif dengan hasil perhitungannya sebanyak -1,051. Hasilnya sesuai dengan yang diuji coba pada Python yang dapat dilihat pada Lampiran 7. Bukti Python.

3.2.6 Validasi K-fold Cross Validation

Sama seperti proses klasifikasi *support vector machine*, validasi *k-fold cross validation* juga menggunakan *library scikit-learn.* Dari analisis validitas akan menghasilkan suatu nilai akurasi yang menggambarkan seberapa besar tingkat keakuratan analisis yang telah dilakukan. Data tersebut akan dijadikan sebagai bahan evaluasi tahap akhir. Hal pertama yang dilakukan adalah mendapatkan nilai dari *confusion matrix*, maka akan mendapatkan sebuah nilai dari *accuracy, precision,* dan *recall* dengan rumus sebagai berikut :

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

3.2.7 Visualisasi Data dengan *Word Cloud*

Pada tahap visualisasi data *word cloud*, *library word cloud* digunakan untuk menghasilkan kata-kata yang paling sering muncul dalam teks yang dianalisis sebelumnya. *word cloud* digunakan untuk visualisasi berdasarkan emosi negatif dan positif. Dalam penelitian ini, Python digunakan untuk memvisualisasikan *word cloud*. Untuk menghasilkan *word cloud*, data kolom *tweet* masyarakat dari proses klasifikasi SVM dengan menggunakan *library word cloud* Python, digunakan untuk memproses data dan menghasilkan gambar yang ditunjukkan di bawah ini. Semakin besar jumlah kata, semakin sering kata tersebut muncul/digunakan dalam teks.

3.2.8 Visualisasi Data dengan Diagram *Pie*

Pada tahap visualisasi, diagram *pie* menggunakan *library matplotlib* untuk menampilkan persentase sentimen positif dan negatif yang merupakan hasil dari sentimen PPKM Mikro yang diberikan oleh pengguna. Data yang digunakan untuk membuat grafik merupakan hasil dari proses klasifikasi SVM karena sudah memiliki label positif dan negatif. Untuk membuat grafik, digunakan nilai jumlah data dan jumlah tanggapan yang dianggap positif atau negatif, yang dapat diperoleh dengan menghitung jumlah tanggapan positif dan negatif. Kemudian gunakan *library matplotlib* di python untuk memproses data untuk membuat diagram *pie*.

3.3 Tahap Akhir

Dalam tahap akhir menggunakan 2 proses yaitu kesimpulan dan saran berdasarkan hasil penelitian yang dilakukan.

3.3.1 Kesimpulan dan Saran

Berdasarkan hasil analisis yang dilakukan maka diambil kesimpulan dari tiap-tiap langkah yang telah dikerjakan. Kemudian diberikan saran kepada penelitian ini untuk rekomendasi agar penelitian menjadi lebih baik lagi. Berdasarkan hasil analisis yang dilakukan maka dapat ditarik kesimpulan dari setiap tahapan yang telah dilakukan. Kemudian memberikan saran terhadap penelitian ini.

BAB IV
HASIL DAN PEMBAHASAN

Berdasarkan data yang telah diambil dari Twitter yang memuat *tweet* dari pengguna mengenai PPKM Mikro. Maka ada bab ini akan dibahas hasil analisa berdasarkan pengolahan data yang telah dilakukan.

4.1 Pelabelan

Pelabelan dilakukan dengan melabeli data secara manual sebanyak 4066 cuitan dengan bantuan 2 sukarelawan. Pada tahapan ini dilakukan pemberian label negatif dan angka label positif pada label data training. Pelabelan ini dikerjakan dengan perangkat pembantu yaitu Microsoft Excel 2016. Pelabelan dilakukan dengan melihat kesimpulan dari suatu kalimat bukan hanya melihat dari sebuah kata. Berikut adalah hasil pelabelan yang ditampilkan pada Gambar 4. 1.

	Text	Id	Label
4059	PPKM Mikro Efektif, Penanganan Covid-19 Indonesia Lebih Baik Dibandingkan Gl	13799472	positif
4060	Secara Grafik Positif Covid19 di PPKM Mikro Desa masih selalu naik setelah libur	13799470	negatif
4061	WASPADA !Grafik Positif Covid 19 di PPKM mikro Masih selalu naik setelah libur l	13799470	negatif
4062	#DariDesaUntukIndonesia...Grafik positif covid-19 di PPKM mikro desa terus mei	13799469	negatif
4063	Grafik positif covid 19 di PPKM Mikro Desa masih slalu naik stelah libur pnjangW	13799466	negatif
4064	Grafik Positif Covid-19 di PPKM Mikro Desa masih selalu naik setelah libur panjar	13799466	negatif
4065	WASPADA! Grafik positif Covid-19 di PPKM Mikro.Tetap jaga prokes.@halimiskan	13799465	negatif
4066	@PadliSugianto: Grafik Positif Covid-19 di PPKM Mikro Desa Masih Selalu Naik S	13799465	negatif
4067	Grafik Positif Covid-19 di PPKM Mikro Desa Masih Selalu Naik Setelah Libur Pania	13799464	negatif

Gambar 4. 1 Hasil Pelabelan

4.2 Text Preprocessing

Pada Tahap text preprocessing menggunakan Google Collaboratory Python Sebelum masuk ke dalam tahap text preprocessing data terlebih dimasukkan ke dalam Google Collaboratory Python. Langkah pertama adalah memasukkan *file* excel ke dalam *directory python* yang tersedia pada Google Collaboratory Python. Berikut adalah *source code* memasukkan data yang ditampilkan pada Gambar 4. 2. Dengan dilakukan memanggil *library pandas* yang dinamai sebagai *pd* untuk mempermudah penulisan. Setelah itu memasukkan *file ppkmmikroperhtengahanapril.xlsx* kedalam direktori python dan memanggil *file* tersebut dengan *pd* yang disimpan ke dalam *dataframe* yang merupakan *array* 2 dimensi dengan nama *df_preprocessed*. Lalu *dataframe* tersebut dihapus kolom yang bernamakan "*id*" dengan fungsi *drop,* dan disimpan ke dalam

df_preprocessed. Lalu mencetak *df_preprocessed*. Hasil dari data akan digunakan pada tahap *text preprocessing* dapat dilihat Gambar L2. 1. pada Lampiran 2. Text Preprocessing.

```
import pandas as pd

df_preprocessed = pd.read_excel("/content/ppkmmikropertengahanapril.xlsx")
df_preprocessed = df_preprocessed.drop(columns=['Id'])
df_preprocessed
```

Gambar 4. 2 *Source code* Memasukkan Data

Berikut adalah source code perubahan label yang ditampilkan pada Gambar 4. 3. Yang pertama adalah membuat data list dengan nama Label. Lalu dilakukan perulangan *index (*indeks baris*)* pada baris yang terdapat pada *df_preprocessed* dengan fungsi *iterrows* karena mengembalikan hasil label untuk setiap baris. Setelah itu dilakukan percabangan baris dari kolom Label, apabila positif maka *datalist* Label akan dinamai angka 1 dengan fungsi *append*(item). Apabila negatif maka *datalist* Label akan dinamai angka 0 dengan fungsi *append*(item). Lalu *datalist* Label disimpan ke dalam *dataframe df_preprocessed* dan dicetak. Merubah huruf menjadi numerik guna proses *k-fold cross validation.* Hasil perubahan label dapat dilihat Gambar L2. 2. pada Lampiran 2. Text Preprocessing.

```
Label = []
for index, row in df_preprocessed.iterrows():
    if row["Label"] == "positif":
        Label.append(1)
    else:
        Label.append(0)

df_preprocessed["Label"] = Label
df_preprocessed
```

Gambar 4. 3 *Source Code* Perubahan Label

Berikut adalah *source code* memanggil fungsi *cleansing* yang ditampilkan pada Gambar 4. 4. Pada *python* proses *cleansing* memanggil fungsi *string* dan *re* dengan membuat fungsi *cleansing* dan parameter data. Fungsi *re.sub* digunakan untuk mengubah *link*, *mentions*, RT, angka, baris baru dan spasi berlebih menjadi spasi dengan menggunakan parameter data. Untuk fungsi *string.punctuation* disimpan ke dalam *corpus remove* lalu *corpus* tersebut dipanggil menggunakan fungsi *str.maketrans* untuk membuat perintah untuk mengubah tanda baca menjadi spasi yang disimpan ke dalam *corpus* translator sehingga parameter data memanggil *corpus* translator fungsi *translate.* Lalu parameter dikembalikan.

```python
import string, re

def cleansing(data):
    # remove link
    data = re.sub(r'https?:\/\/\S+',r'', data)

    # removed @mentions
    data = re.sub(r'@[A-Za-z0-9]+',r'', data)

    #removing RT
    data = re.sub(r'RT[\s]+',r'', data)

    # remove number
    data = re.sub(r'[0-9]+',r'', data)

    # remove punctuation
    remove = string.punctuation
    translator = str.maketrans(remove, ' '*len(remove))
    data = data.translate(translator)

    # remove newline
    data = data.replace('\n', ' ')

    # remove space
    data = data.replace('[\s]+','')

    return data
```

Gambar 4. 4 *Source code* Fungsi *Cleansing*

Setelah membuat fungsi maka melakukan pemanggilan fungsi dengan cara membuat *datalist tweet* lalu membuat perulangan *index* dengan baris pada *df_preprocessed*, *datalist* diisi menggunakan append dengan fungsi *cleansing* untuk tiap baris dari *array* "*Text*". Lalu *datalist* tersebut disimpan ke dalam *df_preprocessed* pada *array "Text"*. Setelah itu dilakukan mencetak *dataframe df_preprocessed* yang dapat dilihat pada Gambar 4. 5. Hasil tahap *cleansing* dapat dilihat Gambar L2. 3. pada Lampiran 2. Text Preprocessing.

```python
#cleansing
tweet = []
for index, row in df_preprocessed.iterrows():
    tweet.append(cleansing(row["Text"]))

df_preprocessed["Text"] = tweet
df_preprocessed.head()
```

Gambar 4. 5 *Source Code* Memanggil Fungsi *Cleansing*

Setelah dilakukan *cleansing*, selanjutnya adalah tahap *case folding*. Pada *python* proses *case folding* dilakukan menggunakan fungsi *string*. Dilakukan dengan mengubah *df_preprocessed array "Text"* menjadi *string* semua dengan menggunakan fungsi *astype*(*str*) dan mengubahnya menjadi huruf kecil (*lowercase*) dengan menggunakan fungsi *str.lower*(), lalu disimpan ke dalam *df_preprocessed "Text"* dan dicetak. Berikut adalah *source code* membuat fungsi *case folding* yang ditampilkan pada Gambar 4. 6. Sedangkan hasil tahap *case folding* dapat dilihat Gambar L2. 4. pada Lampiran 2. Text Preprocessing.

```
#casefolding
df_preprocessed["Text"] = df_preprocessed["Text"].astype(str).str.lower()
df_preprocessed.head()
```

Gambar 4. 6 *Source Code Case Folding*

Setelah dilakukan *case folding*, selanjutnya adalah tahap *tokenizing.* Pada *python* proses *tokenizing* dilakukan menggunakan library *nltk_tokenize* dengan fungsi *nltk* dan *TweetTokenizer.* Membuat fungsi *tokenizing* dengan fungsi *TweetTokenizer* dan pengaturan seperti gambar dan disimpan ke dalam *tokenizer*. Lalu *tokenizer* dipanggil menggunakan fungsi *tokenize* dengan parameter data dan disimpan ke dalam data, lalu parameter dikembalikan. Berikut adalah *source code* membuat fungsi *tokenizing* yang ditampilkan pada Gambar 4. 7. Sedangkan hasil tahap *cleansing* dapat dilihat Gambar L2. 4. pada Lampiran 2. Text Preprocessing.

```
import nltk

#tokenizing import
from nltk.tokenize import TweetTokenizer

def tokenizing(data):
  tokenizer = TweetTokenizer(preserve_case=False, strip_handles=True, reduce_len=True)
  data = tokenizer.tokenize(data)

  return data
```

Gambar 4. 7 *Source Code* Membuat Fungsi *Tokenizing*

Setelah membuat fungsi *tokenizing* maka selanjutnya adalah memanggil fungsi tersebut, membuat *datalist tweet* lalu membuat perulangan *index* dengan baris pada *df_preprocessed*, *datalist* diisi menggunakan *append* dengan fungsi *tokenizing* untuk tiap baris dari *array* "*Text"*. Lalu *datalist* tersebut disimpan ke dalam *df_preprocessed* pada *array "Text"*. Setelah itu dilakukan mencetak *dataframe df_preprocessed* dengan fungsi *head*() untuk mencetak 5 data dari atas. Berikut adalah *source code* memanggil fungsi *tokenizing* yang ditampilkan pada Gambar 4. 8. Sedangkan Hasil tahap *tokenizing* dapat dilihat Gambar L2. 5. pada Lampiran 2. Text Preprocessing.

```
#tokenizing
tweet = []
for index, row in df_preprocessed.iterrows():
    tweet.append(tokenizing(row["Text"]))

df_preprocessed["Text"] = tweet
df_preprocessed.head()
```

Gambar 4. 8 *Source Code* Memanggil Fungsi *Tokenizing*

Setelah dilakukan *tokenizing*, selanjutnya adalah tahap *stopword removal* namun karena pada proses *stopword removal* hanya bisa menerima huruf oleh karena itu tanda baca harus dihapus. Dengan cara membuat fungsi hapus_tandabaca dengan parameter data, lalu data akan diubah menjadi spasi dengan fungsi *join* untuk melakukan perulangan *char* atau kata terhadap *string.pucntuation.* Setelah membuat fungsi hapus_tandabaca maka selanjutnya adalah memanggil fungsi tersebut, membuat *datalist tweet* lalu membuat perulangan *index* dengan baris pada *df_preprocessed*, *datalist* diisi menggunakan *append* dengan fungsi hapus_tandabaca untuk tiap baris dari *array* "*Text".* Lalu *datalist* tersebut disimpan ke dalam *df_preprocessed* pada *array "Text".* Setelah itu dilakukan mencetak *dataframe df_preprocessed* dengan fungsi *head*() untuk mencetak 5 data dari atas. Berikut adalah *source code* menghapus tanda baca fungsi *tokenizing* yang ditampilkan pada Gambar 4. 9. Sedangkan Hasil tahap menghapus tanda baca *tokenizing* dapat dilihat Gambar L2. 6. pada Lampiran 2. Text Preprocessing.

```
def hapus_tandabaca(data):
  data=' '.join([char for char in data if char not in string.punctuation])
  return data

tweet = []
for index, row in df_preprocessed.iterrows():
    tweet.append(hapus_tandabaca(row["Text"]))

df_preprocessed["Text"]= tweet
df_preprocessed.head()
```

Gambar 4. 9 *Source Code* Menghapus Tanda Baca *Tokenizing*

Selanjutnya adalah tahap *stopword removal* bertujuan menghapus kata-kata yang tidak penting dalam deskripsi dengan memeriksa apakah kata-kata hasil *parsing* dalam deskripsi yang termasuk dalam daftar kata tidak penting (*stopword*). Pada *python* proses *stopword* dengan mengunduh package *nltk stopword* dan menggunakan *library Sastrawi* dengan fungsi *StopWordRemoverFactory.* Namun untuk menggunakan *library sastrawi* harus menggunakan PIP atau program manajemen paket. Setelah itu memanggil *library Sastrawi.StopWord.Remover.StopWordRemoverFactory* dan memanggil fungsi *StopWordRemoverFactory*. Lalu menyimpan fungsi *StopWordRemoverFactory* ke dalam *corpus factory.* Lalu *corpus* tersebut dipanggil menggunakan fungsi *create_stop_word_remover* dan disimpan ke dalam *stopword.* Setelah membuat *corpus stopword* yang berisikan fungsi maka selanjutnya adalah memanggil fungsi

tersebut, membuat *datalist tweet* lalu membuat perulangan *index* dengan baris pada *df_preprocessed*, *datalist* diisi menggunakan append dengan fungsi *stopword* untuk tiap baris dari *array* "*Text"*. Lalu *datalist* tersebut disimpan ke dalam *df_preprocessed* pada *array "Text"*. Setelah itu dilakukan mencetak *dataframe df_preprocessed* dengan fungsi *head*() untuk mencetak 5 data dari atas. Berikut adalah *source code stopword removal* yang ditampilkan pada Gambar 4. 10. Sedangkan Hasil tahap *stopword removal* dapat dilihat Gambar L2. 7. pada Lampiran 2. Text Preprocessing.

```
pip install PySastrawi

nltk.download('stopwords')

# import library stopword
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory

factory = StopWordRemoverFactory()
stopword = factory.create_stop_word_remover()

tweet = []
for index, row in df_preprocessed.iterrows():
    tweet.append(stopword.remove(row["Text"]))

df_preprocessed["Text"] = tweet
df_preprocessed.head()
```

Gambar 4. 10 *Source Code Stopword Removal*

Setelah dilakukan *stopword removal*, selanjutnya adalah tahap *stemming* bertujuan untuk mengubah bentuk kata menjadi akar kata dasar sesuai dengan struktur morfologi bahasa Indonesia yang baik dan benar. Pada *python* proses *stemming* dilakukan menggunakan *library Sastrawi* dengan fungsi *StemmerFactory*. Hal pertama yang dilakukan adalah memanggil *library Sastrawi.Stemmer.StemmerFactory* dan memanggil fungsi *StemmerFactory*. lalu menyimpan fungsi S*temmerFactory* ke dalam *corpus factory*. Lalu *corpus* tersebut dipanggil menggunakan fungsi *create_stemmer* dan disimpan ke dalam *stemmer*. Setelah membuat *corpus stemmer* yang berisikan fungsi maka selanjutnya adalah memanggil fungsi tersebut, membuat *datalist tweet* lalu membuat perulangan *index* dengan baris pada *df_preprocessed*, *datalist* diisi menggunakan *append* dengan fungsi *stemmer* untuk tiap baris dari *array* "*Text"*. Lalu *datalist* tersebut disimpan ke dalam *df_preprocessed* pada *array "Text"*. Setelah itu dilakukan mencetak *dataframe df_preprocessed* dengan fungsi *head*() untuk mencetak 5 data dari atas. Berikut adalah *source code stemming* yang ditampilkan pada Gambar 4. 11.

Sedangkan Hasil tahap *stemming* dapat dilihat Gambar L2. 8. pada Lampiran 2. Text Preprocessing.

```
# stemming
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
factory = StemmerFactory()
stemmer = factory.create_stemmer()

tweet = []
for index, row in df_preprocessed.iterrows():
    tweet.append(stemmer.stem(row["Text"]))

df_preprocessed["Text"] = tweet
df_preprocessed.head()
```

Gambar 4. 11 *Source Code Stemming*

4.3 Pembagian Data

Hasil *crawling* data dari Twitter mengenai cuitan ppkm mikro dengan menggunakan Bahasa Indonesia sebanyak 4066 cuitan dilakukan pembagian data dengan membagi data menjadi data training sebanyak 70% dan data testing 30%. Pembagian data pada *python* menggunakan *library sklearn* dan fungsi *train_test_split*. Yang pertama dilakukan adalah memanggil *library sklearn.model_selection* lalu memanggil fungsi *train_test_split*. Sehingga fungsi tersebut dapat digunakan untuk membagi data *array text* dan *array* label pada *dataframe df_preprocessed* dengan ukuran sesuai dengan *test_size* yang menunjukkan 0,3 atau 30% dimana itu adalah persentase untuk data *testing* dan untuk data *training* otomatis menggunakan sisanya. Untuk parameternya sendiri menggunakan *stratify* guna pemisahan yang dihasilkan proporsinya tetap sama. Karena pada setiap menjalankan data akan mengacak data sehingga hasilnya berbeda maka dibutuhkan *random_state* untuk kode produksi agar setiap kali menjalankan progam tetap menghasilkan hasil yang sama. Setelah itu disimpan ke dalam 4 *corpus* yaitu *X_train, X_test, y_train,* dan *y_test,* lalu cetak keempat *corpus* tersebut. *X_train* merupakan data *training, X_test* merupakan data *testing, y_train* merupakan label data *training,* dan *y_test* merupakan label data *testing*. Berikut adalah *source code* pembagian data yang ditampilkan pada Gambar 4. 12. Hasil data *training* dapat dilihat Gambar L3. 1. pada Lampiran 3. Pembagian Data. Sedangkan Hasil data *testing* dapat dilihat Gambar L3. 2. pada Lampiran 3. Pembagian Data.

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df_preprocessed['Text'], df_preprocessed['Label'],
                                                    test_size=0.3, stratify=df_preprocessed['Label'], random_state=30)
X_train, X_test, y_train, y_test
```

Gambar 4. 12 *Source Code* Pembagian Data

Namun sebelum memasuki proses pembobotan *term frequency-inverse document frequency* dilakukan penyalinan data *testing* dari data *X_test* dan data keseluruhan yang berisikan 4066 data atau *f_test* dari *dataframe df_preprocessed array Text* yang nantinya akan digabungkan dengan label hasil klasifikasi. Berikut adalah *source code* untuk menyalin data *testing* dan data *crawling* yang ditampilkan pada Gambar 4. 13.

```
data_testing = X_test.copy()
f_test = df_preprocessed['Text']
```

Gambar 4. 13 *Source Code* Salin Data

4.4 Pembobotan Term Frequency-Inverse Document Frequency

Tahap pembobotan *term frequency-inverse document frequency* melakukan pengolahan terhadap data *training* yang bertujuan untuk menghitung bobot setiap kata yang paling umum digunakan pada *information retrieval.* Pembobotan data pada *python* menggunakan *library sklearn* dan fungsi *TfidfVectorizer*. Langkah pertama yang dilakukan adalah memanggil *library sklearn.feature_extraction.text* dan fungsi *TfidfVectorizer*. Lalu memasukkan fungsi tersebut ke dalam *corpus vectorizer*. Sehingga fungsi *vectorizer.fit_transform* dapat digunakan pada data *X_train* dan disimpan ke dalam *corpus x_train* atau data *training.* Lalu fungsi *vectorizer.transform* dapat digunakan pada data *X_test* dan disimpan ke dalam *corpus x_test.* Fungsi *vectorizer.fit_transform* dapat digunakan pada data *f_test* dan disimpan ke dalam *corpus f_test.* Setelah itu dilakukan cetak kedapat 3 *corpus* tersebut dengan fungsi *shape* untuk mengetahui banyaknya kata yang terkandung dalam data tersebut. Setelah itu mencetak *x_train* atau data *training* dengan perintah *print*(). Berikut adalah *source code* pembobotan TF-IDF yang ditampilkan pada Gambar 4. 17. Sedangkan Hasil tahap pembobotan *term frequency-inverse document frequency* dapat dilihat Gambar L4. 1. pada Lampiran 4. Pembobotan Term Frequency-Inverse Document Frequency.

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer()

#tfidf
x_train = vectorizer.fit_transform(X_train)
x_test = vectorizer.transform(X_test)
f_test = vectorizer.transform(f_test)

print(x_train.shape)
print(x_test.shape)
print(f_test.shape)
print(x_train)
```

Gambar 4. 14 *Source Code* Pembobotan TF-IDF

4.5 Klasifikasi Support Vector Machine

Support Vector Machine (SVM) Metode klasifikasi yang digunakan pembelajaran mesin (*supervised learning*) adalah memprediksi kategori berdasarkan model atau pola hasil dari proses *training*. Klasifikasi data pada *python* menggunakan *package XlsxWriter* dengan *library sklearn* dan fungsi *svm, sys,* dan *numpy*. Yang pertama dilakukan adalah memanggil *library sklearn* dan fungsi *svm, sys,* dan *numpy*. sehingga fungsi tersebut dapat digunakan untuk memanggil fungsi *SVC* dengan *kernel linear* dan disimpan ke dalam *corpus clf*. Lalu *x_train* dan *y_train* dipanggil dengan menggunakan fungsi *clf.fit* yang memiliki metode perhitungan bobot terbaik dengan melatih algoritma menggunakan data *training* atau *x_train* dan label data *training* atau *y_train*, juga untuk menghitung metode dan menyimpannya sebagai objek internal. Model tersebut akan digunakan untuk mengklasifikasikan data *testing* atau *x_test* ke negatif atau positif dengan menggunakan fungsi *predict* dan simpan ke dalam *predict*. Setelah itu dilakukan juga hal yang sama terhadap data keseluruhan atau *f_test* dengan menggunakan fungsi *predict* dan simpan ke dalam *f_pred_test,* yang nantinya akan digunakan untuk visualisasi diagram *pie.* Lalu setelah mengklasifikan maka mencetak data, namun sebelum mencetak data mengatur terlebih dahulu pengaturan cetak agar dapat melihat seluruh hasil data dengan menggunakan fungsi *numpy.set_printoption* dan fungsi *max.size* disimpan ke dalam *threshold.* Berikut adalah *source code* klasifikasi *support vector machine* yang ditampilkan pada Gambar 4. 18.

```
#svm
from sklearn import svm
import sys
import numpy

clf = svm.SVC(kernel="linear")
clf.fit(x_train,y_train)
predict = clf.predict(x_test)
f_pred_test = clf.predict(f_test)
numpy.set_printoptions(threshold=sys.maxsize)
print(predict)
len(predict)
```

Gambar 4. 15 Source Code Support Vector Machine

Setelah dilakukan klasifikasi maka proses selanjutnya adalah menyiapkan data sebelum disimpan ke dalam excel. Langkah pertama adalah menyalin data_*testing* menggunakan fungsi *pd* ke dalam *dataframe df_datatesting* dan menyalin predict atau hasil klasifikasi data testing menggunakan fungsi *pd* ke dalam *dataframe* df_*Labeltesting*. Yang pertama adalah membuat data *list* dengan nama Label. Lalu dilakukan perulangan *index (*indeks baris*)* pada baris yang terdapat pada *df_Labeltesting* dengan fungsi *iterrows* karena mengembalikan hasil label untuk setiap baris. Setelah itu dilakukan percabangan baris dari kolom Label, apabila 1 maka *datalist* Label akan dinamai positif dengan fungsi *append*(item). Apabila 0 maka *datalist* Label akan dinamai negatif dengan fungsi *append*(item). Untuk memudahkan membaca data pada excel nantinya. Lalu *datalist* Label disimpan ke dalam *dataframe df_Labeltesting* Lalu *dataframe* tersebut dihapus array yang tidak memiliki judul atau bernamakan "*0*" dengan fungsi *drop*. Setelah itu dilakukan mencetak *dataframe df_Labeltesting* dengan fungsi *head*() untuk mencetak 5 data dari atas. Berikut adalah *source code* perubahan label hasil yang ditampilkan pada Gambar 4. 16.

```
df_datatesting = pd.DataFrame(data_testing)
df_Labeltesting = pd.DataFrame(predict)

Label = []
for index, row in df_Labeltesting.iterrows():
    if row[0] == 1:
        Label.append("positif")
    else:
        Label.append("negatif")

df_Labeltesting["Label"] = Label
df_Labeltesting = df_Labeltesting.drop(columns=[0])
df_Labeltesting.head()
```

Gambar 4. 16 *Source Code* Perubahan Label Klasifikasi Data Testing

Setelah itu untuk menyimpan 2 array ke dalam excel dibutuhkan modul *XlsxWriter* untuk memasang modul tersebut harus menggunakan PIP atau program manajemen paket. Hal pertama yang dilakukan adalah menggunakan fungsi pd.ExcelWriter

untuk membuat file excel yang memiliki nama dan format *hasildatatesting.xlsx* dengan mesin *xlsxwriter* dan disimpan ke dalam *corpus writer*. Lalu menyimpan *df_datatesting* ke dalam excel dengan fungsi *to_excel* dengan fungsi *writer* sebagai file excel, dengan *sheet_name* hasilklasifikasi, baris dimulai dari 0, dan kolom dimulai dari 0. Setelah itu menyimpan df_*labeltesting* ke dalam excel dengan fungsi *to_excel* dengan fungsi *writer* sebagai file excel yang sama, dengan *sheet_name* sama yaitu hasilklasifikasi, baris dimulai dari 0, dan kolom dimulai dari 2. Sehingga dapat dilakukan simpan pada *corpus* writer dengan fungsi *save*(). Setelah disimpan maka langkah selanjutnya yaitu memanggil kembali data tersebut menggunakan fungsi *pd.read*_excel dengan direktori "/*content*/hasildata*testing*.xlsx" dan disimpan ke dalam *dataframe dft*. Karena *dataframe* tersebut masih memiliki kolom yang tidak digunakan maka dilakukan penghapusan kolom. Dengan fungsi *dft.drop* dan memilih judul kolom yaitu 0 dan 2 maka kolom tersebut akan dihapus dan disimpan ke dalam *dft*. Lalu cetak *dataframe dft* tersebut. Berikut adalah *source code* simpan klasifikasi data *testing* yang ditampilkan pada Gambar 4. 17. Sedangkan Hasil tahap klasifikasi *support vector machine* dapat dilihat Gambar L5. 1. pada Lampiran 5. Klasifikasi *Support Vector Machine*.

```
pip install XlsxWriter


#save data testing
writer = pd.ExcelWriter('hasildatatesting.xlsx', engine='xlsxwriter')
df_datatesting.to_excel(writer,sheet_name='hasilklasifikasi',startrow=0 , startcol=0)
df_Labeltesting.to_excel(writer,sheet_name='hasilklasifikasi',startrow=0, startcol=2)
writer.save()

dft = pd.read_excel("/content/hasildatatesting.xlsx")
dft = dft.drop(columns=['Unnamed: 0', 'Unnamed: 2'])
dft
```

Gambar 4. 17 *Source Code* Simpan Klasifikasi Data Testing

4.6 Evaluasi dan Validasi K-fold Cross Validation

Validasi *K-fold Cross Validation* digunakan untuk menghasilkan akurasi *support vector machine* guna mengetahui nilai akurasi yang menggambarkan seberapa besar tingkat keakuratan analisis yang telah dilakukan. Evaluasi dan validasi data pada *python* menggunakan *library sklearn*. Langkah pertama yang dilakukan adalah memanggil *library sklearn.model_selection* dengan fungsi *cross_val_score*. Fungsi tersebut digunakan untuk memanggil fungsi *clf* yang digunakan untuk menghitung *x_train* atau data *training* dan *y_train* atau label dari

data *training* tersebut dengan menggunakan fungsi *cv* untuk membagi *cross validation* menjadi 10 tahapan dan disimpan ke dalam *corpus scores*. Lalu dilakukan cetak hasil *cross validation* dengan cara merubah data yang *array* menjadi *list* untuk dapat menghasilkan hasil berupa persentase dengan fungsi *list* dan *map* untuk menampilkan berapa digit setelah koma, dan dengan menggunakan fungsi *format* untuk memanggil data *scores*. Lalu cetak data *scores* yang akan mengeluarkan 10 hasil tahapan dalam bentuk persen. Setelah itu dilakukan perhitungan rata rata dengan cara memanggil fungsi *clf* yang digunakan untuk menghitung *x_train* atau data training dan y_*train* atau label dari data *training* tersebut dengan menggunakan fungsi *cv* untuk membagi *cross validation* menjadi 10 tahapan dan diakhiri dengan fungsi *mean*() untuk menemukan rata-rata dan disimpan cetak. Berikut adalah *source code k-fold cross validation score* yang dapat dilihat pada Gambar 4. 18. Sedangkan hasil validasi *k-fold cross validation* dapat dilihat pada Tabel 4. 4.

```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(clf, x_train, y_train, cv=10)
scores= list(map('{:.2%}'.format,scores))
print(f'{scores}')
print(f"{cross_val_score(clf, x_train, y_train, cv=10).mean():.2%}")
```

Gambar 4. 18 *Source Code K-fold Cross Validation*

Hal selanjutnya yang dilakukan untuk dijadikan sebagai bahan evaluasi tahap akhir adalah mendapatkan nilai dari *confusion matrix,* dengan memanggil terlebih dahulu *library* dan fungsi yang digunakan pada tahap evaluasi yaitu *library sklearn.metrics* dengan fungsi *recall_score, precision_score, confusion_matrix,* dan *accuracy score*. Lalu dengan menggunakan fungsi confusion_matrix maka dapat mencocokkan antara *y_test* atau label dari data testing dengan *predict* atau label dari hasil klasifikasi data *testing* dan dengan fungsi *ravel* untuk mengurai kedua data tersebut. Setelah itu hasil akan disimpan ke dalam *tp* atau *true positive, fp* atau *false positive, tn* atau *true negative,* dan *fp* atau *false negative*. Berikut adalah *source code confusion matrix* yang dapat dilihat pada Gambar 4. 19. Sedangkan hasil evaluasi *confusion matrix* dapat dilihat pada Gambar 4. 27.

```python
# import library evaluation
from sklearn.metrics import recall_score, precision_score, confusion_matrix, accuracy_score

# confusion matrix
tn, fp, fn, tp = confusion_matrix(y_test, predict).ravel()

# tn
print("Hasil True Negative adalah: ")
print(tn)

# fp
print("Hasil False Positive adalah: ")
print(fp)

# fn
print("Hasil False Negative adalah: ")
print(fn)

# tp
print("Hasil True Positive adalah: ")
print(tp)
```

Gambar 4. 19 *Source Code Confusion Matrix*.

Setelah confusion matrix ditemukan maka akan mendapatkan sebuah nilai dari *accuracy*, *precision*, dan *recall*. Dalam *python* kita bisa menggunakan *recall_score, precision_score,* dan *accuracy_score* dari *library sklearn* untuk proses evaluasi. Untuk mencari *accuracy* menggunakan fungsi *accuracy_score* untuk menghitung *y_test* atau label dari data *testing* dan *predict* atau label dari hasil klasifikasi data *testing* dengan mencetaknya menjadi persen dengan 2 digit dibelakang koma. Untuk mencari *precission* menggunakan fungsi *accuracy_score* untuk menghitung *y_test* atau label dari data *testing* dan *predict* atau label dari hasil klasifikasi data *testing* dengan mencetaknya menjadi persen dengan 2 digit dibelakang koma. Untuk mencari *recall* menggunakan fungsi *accuracy_score* untuk menghitung *y_test* atau label dari data *testing* dan *predict* atau label dari hasil klasifikasi data *testing* dengan mencetaknya menjadi persen dengan 2 digit dibelakang koma. Berikut adalah *source code accuracy precission,* dan *recall* yang dapat dilihat pada Gambar 4. 20. Sedangkan hasil *accuracy precission,* dan *recall* dapat dilihat pada Gambar 4. 28.

```python
# accuracy score
print("accuracy score hasil prediksi adalah: ")
print(f"{accuracy_score(y_test, predict):.2%}")

# precision score
print("precision score hasil prediksi adalah: ")
print(f"{precision_score(y_test, predict):.2%}")

# recall score
print("recall score hasil prediksi adalah: ")
print(f"{recall_score(y_test, predict):.2%}")
```

Gambar 4. 20 Accuracy, Precision, dan Recall.

4.7 Visualisasi Data dengan *Word Cloud*

Setelah dilakukan klasifikasi maka proses selanjutnya adalah menyiapkan data sebelum disimpan ke dalam excel. Langkah pertama adalah menyalin *df_preprocessed array "Text"* menggunakan fungsi *pd* ke dalam *dataframe df_datafull* dan menyalin *f_pred_test* atau hasil klasifikasi seluruh data menggunakan fungsi *pd* ke dalam *df_Labelfull*. Lalu *df_Labelfull* disimpan kedalam *df_Labelfull* array "Label" guna pengelompokkan pada diagram *pie*. Lalu *dataframe* tersebut dihapus array yang tidak memiliki judul atau bernamakan "*0*" dengan fungsi *drop*. Berikut adalah *source code* hasil klasifikasi seluruh data yang ditampilkan pada Gambar 4. 21.

```
df_datafull = df_preprocessed['Text']
df_Labelfull = pd.DataFrame(f_pred_test)
df_Labelfull["Label"] = df_Labelfull
df_Labelfull = df_Labelfull.drop(columns=[0])
```

Gambar 4. 21. *Source Code* Hasil Klasifikasi Seluruh Data

Setelah itu untuk menyimpan 2 array ke dalam excel dibutuhkan modul *XlsxWriter*. Hal pertama yang dilakukan adalah menggunakan fungsi *pd.ExcelWriter* untuk membuat file excel yang memiliki nama dan format *hasildatatesting.xlsx* dengan mesin *xlsxwriter* dan disimpan ke dalam *corpus writer*. Lalu menyimpan *df_datafull* ke dalam excel dengan fungsi *to_excel* dengan fungsi *writer* sebagai file excel, dengan *sheet_name* "hasilklasifikasi", baris dimulai dari 0, dan kolom dimulai dari 0. Setelah itu menyimpan *df_Labelfull* ke dalam excel dengan fungsi *to_excel* dengan fungsi *writer* sebagai file excel yang sama, dengan *sheet_name* sama yaitu "datafull", baris dimulai dari 0, dan kolom dimulai dari 2. Sehingga dapat dilakukan simpan pada *corpus writer* dengan fungsi *save()*. Setelah disimpan maka langkah selanjutnya yaitu memanggil kembali data tersebut menggunakan fungsi *pd.read_excel* dengan direktori "/*content*/hasildata*testing*.xlsx" dan disimpan ke dalam *dataframe dft*. Karena *dataframe* tersebut masih memiliki kolom yang tidak digunakan maka dilakukan penghapusan kolom. Dengan fungsi *dft.drop* dan memilih judul kolom yaitu 0 dan 2 maka kolom tersebut akan dihapus dan disimpan ke dalam *dft*. Dilakukan dengan

mengubah *dft array "Text"* menjadi *string* semua dengan menggunakan fungsi *astype(str)*.

```
#full data
writer = pd.ExcelWriter('datatweet.xlsx', engine='xlsxwriter')
df_datafull.to_excel(writer,sheet_name='datafull',startrow=0 , startcol=0)
df_Labelfull.to_excel(writer,sheet_name='datafull',startrow=0 , startcol=2)
writer.save()

dft = pd.read_excel("/content/datatweet.xlsx")
dft = dft.drop(columns=['Unnamed: 0', 'Unnamed: 2'])
dft["Text"] = dft["Text"].astype(str)
```

Gambar 4. 22. *Source Code* Simpan Klasifikasi Seluruh Data

Pada tahap visualisasi data dengan *word cloud* menggunakan library *wordcloud* dengan fungsi *WordCloud* dan *library matplotlib.pyplot* sebagai *plt*. Hal pertama yang dilakukan adalah apabila data dalam Label bernilai 0 maka akan masuk kedalam *corpus train_s0.* Lalu menggabungkan seluruh kata yang terdapat pada *corpus* menggunakan fungsi *join* dan *for*, disimpan ke dalam *corpus all_text_0*. Membuat *stopword* ppkm dan mikro karena kata tersebut sudah pasti muncul paling banyak karena merupakan kata kunci pada setiap *tweet.* Lalu membuat *word cloud* dengan fungsi *WordCloud* yang berisikan fungsi *stopword* dengan memanggil *corpus stop_words*, fungsi *colormap* untuk mengubah warna huruf, mengatur lebar dan tinggi *layout* dengan menggunakan fungsi *width* dan *height*, mengatur *mode* warna, mengatur *background_color*, dengan menggunakan *corpus all_text_s0*. Membuat *figure layout* menggunakan fungsi *plt.figure*. digunakan fungsi *interpolation* untuk mengatur besar kecilnya dengan menggunakan perhitungan interpolasi bilinear. Lalu *axis off* bertujuan untuk membuat tumpukan data namun tidak bersama *index* ataupun kolom. Mengatur *margin* pada sumbu x dan y, lalu cetak menggunakan perintah *show*(). Berikut adalah *source code word cloud* negatif yang ditampilkan pada Gambar 4. 23. Hasil diagram *word cloud* negatif dapat dilihat Gambar L4. 1. Diagram *Pie* pada Lampiran 4. Visualisasi Data dengan word Cloud.

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

#negative
train_s0 = dft[dft["Label"] == 0]
all_text_s0 = ' '.join(word for word in train_s0["Text"])
stop_words = ["ppkm", "mikro"]
wordcloud = WordCloud(stopwords= stop_words, colormap='Reds',
                      width=1000, height=1000, mode='RGBA', background_color='white').generate(all_text_s0)
plt.figure(figsize=(20,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.margins(x=0, y=0)
plt.show()
```

Gambar 4. 23 *Word Cloud* Negatif

Pada tahap visualisasi data dengan *word cloud* menggunakan library *wordcloud* dengan fungsi *WordCloud* dan *library matplotlib.pyplot* sebagai *plt*. Hal pertama yang dilakukan adalah apabila data dalam Label bernilai 0 maka akan masuk kedalam *corpus train_s0.* Lalu menggabungkan seluruh kata yang terdapat pada *corpus* menggunakan fungsi *join* dan *for*, disimpan ke dalam *corpus all_text_0*. Membuat *stopword* ppkm dan mikro karena kata tersebut sudah pasti muncul paling banyak karena merupakan kata kunci pada setiap *tweet.* Lalu membuat *word cloud* dengan fungsi *WordCloud* yang berisikan fungsi *stopword* dengan memanggil *corpus stop_words*, fungsi *colormap* untuk mengubah warna huruf, mengatur lebar dan tinggi *layout* dengan menggunakan fungsi *width* dan *height*, mengatur *mode* warna, mengatur *background_color*, dengan menggunakan *corpus all_text_s0*. Membuat *figure layout* menggunakan fungsi *plt.figure*. digunakan fungsi *interpolation* untuk mengatur besar kecilnya dengan menggunakan perhitungan interpolasi bilinear. Lalu *axis off* bertujuan untuk membuat tumpukan data namun tidak bersama *index* ataupun kolom. Mengatur *margin* pada sumbu x dan y, lalu cetak menggunakan perintah *show*(). Berikut adalah *source code word cloud* positif yang ditampilkam pada Gambar 4. 24. Hasil diagram *word cloud* positif dapat dilihat Gambar L4. 2. Diagram *Pie* pada Lampiran 4. Visualisasi Data dengan word Cloud.

```
#positive
train_s1 = dft[dft["Label"] == 1]
all_text_s1 = ' '.join(word for word in train_s1["Text"])
stop_words = ["ppkm", "mikro"]
wordcloud = WordCloud(stopwords= stop_words, colormap='Blues',
                      width=1000, height=1000, background_color='white', mode='RGBA').generate(all_text_s1)
plt.figure( figsize=(20,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.margins(x=0, y=0)
plt.show()
```

Gambar 4. 24 *Source Code Word Cloud* Positif

4.8 Visualisasi Data dengan Diagram *Pie*

Dft disalin ke dalam *dft_diagram* yang nantinya digunakan pada proses diagram *pie.* Membuat data list dengan nama Label. Lalu dilakukan perulangan *index (*indeks baris*)* pada baris yang terdapat pada *dft_diagram* dengan fungsi *iterrows* karena mengembalikan hasil label untuk setiap baris. Setelah itu dilakukan percabangan baris dari kolom Label, apabila angka 1 maka *datalist* Label akan dinamai positif dengan fungsi *append*(item). Apabila angka 0 maka *datalist* Label akan dinamai negatif dengan fungsi *append*(item). Lalu *datalist* Label disimpan ke dalam *dataframe dft_diagram* dan dicetak. Merubah numerik menjadi huruf guna indeks diagram *pie.* Berikut adalah *source code* perubahalan label data diagram yang ditampilkan pada Gambar 4. 25.

```
dft_diagram = dft.copy()
Label = []
for index, row in dft_diagram.iterrows():
    if row["Label"] == 1:
        Label.append("positif")
    else:
        Label.append("negatif")

dft_diagram["Label"] = Label
```

Gambar 4. 25. *Source Code* Perubahalan Label Data Diagram

Pada Gambar 4. 23, merupakan *source code* diagram *pie* dengan dilakukan memanggil *library matplotlib.ticker* yang dinamai sebagai *ticker*, memanggil *library matplotlib.cm* yang dinamai sebagai *cm,* memanggil *library matplotlib* yang dinamai sebagai *mpl,* dan memanggil *library matplotlib.pyplot* yang dinamai sebagai *plt* untuk mempermudah penulisan, serta memanggil *library GridSpec* dengan fungsi *matplotlib.* Langkah pertama yang dilakukan adalah mengelompokkan data array label dengan menggunakan *group by* dan pengumpulan dari perhitungan jumlah karakter yang muncul dengan menggunakan fungsi *agg* dan *count* dan menyimpan kedalam *corpus title_type* sehingga dapat dicetak dengan menggunakan perintah *print()*. Setelah itu melakukan sorting dengan menggunakan *title_type* dengan fungsi *Text.sort_values()* untuk mengambil indeksnya saja yaitu label negatif dan positif lalu disimpan ke dalam *type_labels.* Melakukan sorting dengan menggunakan *title_type* dengan fungsi *Text.sort_values()* saja untuk mengambil hasil pengelompokan yaitu 491 dan 3574 lalu disimpan ke dalam *type_counts*. Setelah itu mengatur ukuran diagram

menggunakan fungsi *figsize*. Dan mengatur *figure layout* dengan menggunakan *GridSpec*. Maka dapat memilih warna dengan fungsi *cmap* juga merubah warna dari diagram, untuk negatif diberi warna *tomato* dan untuk positif diberi warna *lightskyblue*. *plt.subplot* digunakan untuk mengatur *grid* dengan aspek rasionya. Sehingga dapat membuat diagram *pie* yang berisikan label dan jumlahnya dengan persentase 2 digit dibelakang koma, menambahkan bayangan, memanggil warna yang telah dibuat dengan *corpus c1* lalu cetak dengan perintah *show*(). Hasil diagram pie dapat dilihat pada Gambar 4. 29. Visualisasi Data dengan Diagram *Pie*.

```
import matplotlib.ticker as ticker
import matplotlib.cm as cm
import matplotlib as mpl
from matplotlib.gridspec import GridSpec
import matplotlib.pyplot as plt

title_type = dft_diagram.groupby('Label').agg('count')
print(title_type)

type_labels = title_type.Text.sort_values().index
type_counts = title_type.Text.sort_values()

plt.figure(1, figsize=(10,10))
the_grid = GridSpec(2, 2)

cmap = plt.get_cmap('Spectral')
cl = ['tomato','lightskyblue']

plt.subplot(the_grid[1, 1], aspect=1)
type_show_ids = plt.pie(type_counts, labels=type_labels, autopct='%2.1f%%', shadow=True, colors=cl, radius=3)
plt.show()
```

Gambar 4. 26. *Source Code* Diagram *Pie*

4.9 Hasil pembahasan

Tahap pembobotan *term frequency-inverse document frequency* melakukan pengolahan terhadap data *training* yang bertujuan untuk menghitung bobot setiap kata yang paling umum digunakan pada *information retrieval.* sebuah ukuran statistik yang digunakan untuk mengevaluasi seberapa penting sebuah kata di dalam sebuah dokumen atau dalam sekelompok kata. Untuk dokumen tunggal tiap kalimat dianggap sebagai dokumen. Frekuensi dokumen yang mengandung kata tersebut menunjukkan seberapa umum kata tersebut. Kata yang memiliki bobot akan muncul pada hasil pembobotan *term frequency-inverse document frequency* tiap kalimat. Bobot kata semakin besar jika sering muncul dalam suatu dokumen dan semakin kecil jika muncul dalam banyak dokumen. Pembobotan data pada *python*

menggunakan *library sklearn* dan fungsi *TfidfVectorizer*. Berikut adalah hasil dari pembobotan TF-IDF yang ditampilkan pada Gambar 4. 30.

```
(2846, 3625)
(1220, 3625)
(4066, 3625)
  (0, 1870)     0.058772674052201754
  (0, 2781)     0.3565982761355653
  (0, 2433)     0.05875203042496681
  (0, 1607)     0.18808155619493622
  (0, 3054)     0.36430151732755905
  (0, 1535)     0.21794016871704716
  (0, 1256)     0.21633313957436345
  (0, 737)      0.3965849774531933
  (0, 3080)     0.3965849774531933
  (0, 1411)     0.42369883619480736
  (0, 889)      0.3308525909835373
```

Gambar 4. 27. Hasil dari Pembobotan TF-IDF

Support Vector Machine (SVM) Metode klasifikasi yang digunakan pembelajaran mesin (*supervised learning*) adalah memprediksi kategori berdasarkan model atau pola hasil dari proses *training*. Data *training* dan label data *training* digunakan sebagai parameter untuk menghitung metode dan menyimpannya sebagai objek internal. Klasifikasi data pada *python* menggunakan *package XlsxWriter* dengan *library sklearn* dan fungsi *svm*. Model tersebut akan digunakan untuk mengklasifikasikan data *testing* ke negatif dan positif. Sebelum dilakukan klasifikasi data testing harus melalui tahap text preprocessing dan tahap pembobotan *term frequency-inverse document frequency*. Berikut adalah hasil dari klasifikasi *support vector machine* yang ditampilkan pada Gambar 4. 31.

	Text	label
0	polrres pematangsiantar laksana operasi yustis...	positif
1	kspk ipda supandi pimpin operasi yustisi tni p...	positif
2	protokol sehat cegah sebar virus covid edukasi...	positif
3	babinsa koramil tellu siatting serda muh asdar...	positif
4	polres kebumen polsek jajar rutin gelar operas...	positif
...	...	...
1215	kapolda jatim tinjau laksana ppkm mikro dusun ...	positif
1216	giat laksana ppkm skala mikro wilayah kodim pr...	positif
1217	darnramil rambipuji kapten chb mulyadi konfirm...	positif
1218	giat ops yustisi tega disiplin patuh protokol ...	positif
1219	grafik positif covid ppkm mikro desa libur har...	negatif

1220 rows × 2 columns

Gambar 4. 28. Hasil *Test* Data *Testing*

Validasi *K-fold Cross Validation* digunakan untuk menghasilkan akurasi *support vector machine* guna mengetahui nilai akurasi yang menggambarkan seberapa besar tingkat keakuratan analisis yang telah dilakukan. Evaluasi dan validasi data pada *python* menggunakan *library sklearn*. Berikut adalah hasil evaluasi *cross validation score* yang dapat dilihat pada Tabel L4. 4.

Tabel 4. 1. *K-fold Cross Validation*

Fold ke	1	2	3	4	5	6	7	8	9	10	Rata-rata
CV S	97.54 %	95.79 %	97.89 %	95.79 %	95.79 %	96.84 %	96.48 %	96.48 %	95.42 %	96.13 %	96.42 %

Hal selanjutnya yang dilakukan untuk dijadikan sebagai bahan evaluasi tahap akhir adalah mendapatkan nilai dari *confusion matrix.* Dalam *python* bisa menggunakan *library sklearn* dengan fungsi *confusion_matrix*. Evaluasi dengan *confusion matrix* untuk mengetahui hasil klasifikasi *true positive* atau data positif dan diklasifikasikan positif, *false positive* atau data positif yang diklasifikasikan negatif, *true negative* atau data negatif dan diklasifikasikan negatif, dan *false negative* atau data negatif dan diklasifikan positif yang ditampilkan pada Gambar 4. 27.

```
Hasil True Negative adalah:
135
Hasil False Positive adalah:
26
Hasil False Negative adalah:
9
Hasil True Positive adalah:
1050
```

Gambar 4. 29. *Confusion Matrix*.

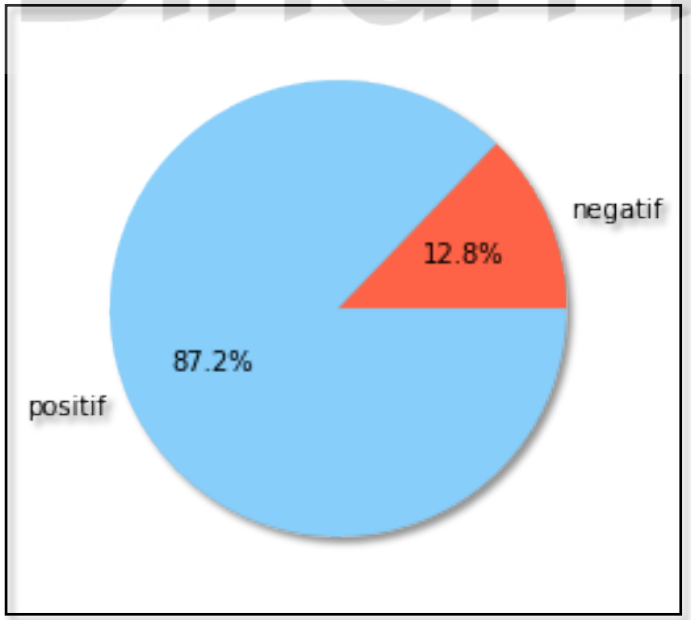
Maka akan mendapatkan sebuah nilai dari accuracy, precision, dan recall yang ditampilkan pada Gambar 4. 28. Dalam *python* kita bisa menggunakan *recall_score, precision_score,* dan *accuracy_score* dari *library sklearn* untuk proses validasi. Hasil prediksi *accuracy* adalah 97,13% yang merupakan persenan sentimen yang benar diprediksi positif dan negatif dari seluruh data *tweet*. Hasil prediksi *precission* adalah 97,58% yang merupakan persenan sentimen yang benar

positif dari keseluruhan data *tweet* yang dipresiksi positif. Hasil prediksi *recall* adalah 99,15% yang merupakan persenan sentimen yang benar negatif dari keseluruhan data *tweet* yang dipresiksi negatif. Hasil prediksi *f1* adalah 98.36 yang merupakan perbandingan rata-rata bobot *precission* dan *recall*. Acuan performansi algoritma yang digunakan adalah *accuracy* karena data tweet memiliki jumlah data *false negative* dan *false positive* yang sangat mendekati.

```
f1 score hasil prediksi adalah:
98.36%
accuracy score hasil prediksi adalah:
97.13%
precision score hasil prediksi adalah:
97.58%
recall score hasil prediksi adalah:
99.15%
```

Gambar 4. 30. Accuracy, Precision, dan Recall.

Pada Gambar 4. 29, menunjukkan persentase sentimen yang sudah dilakukan klasifikasi. Pada tahap visualisasi dengan diagram *pie* menggunakan *library matplotlib*. Persentase keseluruhan sentimen pada *tweet* PPKM Mikro sebanyak untuk negatif 12,8% atau sebanyak 491 cuitan dan untuk positif 87,2% atau sebanyak 3574 cuitan.



Gambar 4. 31. Diagram Pie

BAB V
PENUTUP

Berdasarkan hasil dari penelitian terkait pengerjaan tugas akhir ini. Maka ada bab ini akan dibahas kesimpulan dan saran.

5.1 Kesimpulan

Penelitian ini telah berhasil melakukan analisis sentimen dengan menggunakan metode *support vector machine* dengan kata kunci ppkm mikro. Berdasarkan implementasi dan hasil evaluasi yang telah dilakukan dengan k-fold cross validation, maka dapat disimpulkan bahwa:

1. Hasil evaluasi menggunakan *k-fold cross validation* pada *support vector machine* menunjukkan jumlah data *false negative* dan *false positive* yang sangat dekat (*symmetric*) dengan hanya selisih 15 data, dan hasil pengujian menunjukkan *accuracy* sebanyak 97,13%. maka a*ccuracy* bisa dikatakan sangat baik digunakan sebagai acuan performansi. Ini membuktikan bahwa *Support Vector Machine* (SVM) dapat digunakan untuk melakukan klasifikasi analisis sentimen dengan sangat baik.
2. Hasil visualisasi dari diagram pie menunjukkan hasil analisis sentimen pada *tweet* PPKM Mikro sebanyak 12,8% atau 491 cuitan negatif dan 87,2% atau 3574 cuitan positif. Sehingga dapat diketahui respon masyarakat lebih banyak beranggapan positif terhadap kebijakan pemerintah tersebut. Berdasarkan hasil prediksi tersebut dapat menjadi tolak ukur atau bahan evaluasi bagi pemerintah agar dapat memperpanjang kebijakan tersebut.

5.2 Saran

Dalam analisis sentimen publik terhadap kebijakan ppkm mikro menggunakan algoritma *support vector machine* pada twitter yang telah dibuat, dapat diberikan saran untuk pengembangan penelitian ini sebagai berikut:

1. Mengganti pembagian proporsi penggunaan data *training* dan data *testing*.
2. Menyamakan penggunaan data positif dan negatif pada data *training*.
3. Melakukan kamus tentang bahasa gaul, karena pada sosial media twitter review bahasa Indonesia terlalu banyak bahasa yang kurang baku.

DAFTAR PUSTAKA

Amrizal, V. (2018). Penerapan Metode Term Frequency Inverse Document Frequency (Tf-Idf) Dan Cosine Similarity Pada Sistem Temu Kembali Informasi Untuk Mengetahui Syarah Hadits Berbasis Web (Studi Kasus: Hadits Shahih Bukhari-Muslim). *Jurnal Teknik Informatika*, *11*(2), 149–164. https://doi.org/10.15408/jti.v11i2.8623

Hikmawan, S., Pardamean, A., & Khasanah, S. N. (2020). Sentimen Analisis Publik Terhadap Joko Widodo terhadap wabah Covid-19 menggunakan Metode Machine Learning. *Jurnal Kajian Ilmiah*, *20*(2), 167–176. https://doi.org/10.31599/jki.v20i2.117

Ihsan, M., Roza, E., & Widodo, E. (2019). Analisis Sentimen Twitter terhadap Bom Bunuh Diri di Surabaya 13 Mei 2018 menggunakan Pendekatan Support Vector Machine. *Prisma 2 (2019): 416-426*, *2*, 416–426.

Mochamad Tri Anjasmoros, Istidadi, dan F. M. (2020). *Analisis sentimen aplikasi go-jek menggunakan metode svm dan nbc (studi kasus: komentar pada play store) 1)*. *Ciastech*, 489–498.

Negara, E. S., Andryani, R., & Saksono, P. H. (2016). Analisis Data Twitter: Ekstraksi dan Analisis Data G eospasial. *Jurnal INKOM*, *10*(1), 27. https://doi.org/10.14203/j.inkom.433

Nomleni, P. (2015). Sentiment Analysis Menggunakan Support Vector Machine (Svm). *Seminar Nasional Teknologi Dan Komunikasi 2015*, *2015*(Sentika), 1–8.

Pamungkas, I. T. S. A. (2018). Analisis Sentimen Terhadap Tokoh Publik Menggunakan Algoritma Support Vector Machine (Svm). *Log!K@*, *8*(1), 69–79.

Pristiyanti, R. I., Fauzi, M. A., & Muflikhah, L. (2018). Sentiment Analysis Peringkasan Review Film Menggunakan Metode Information Gain dan K-Nearest Neighbor. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya*, *2*(3), 1179–1186. http://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/1140

Elena, M. (2021, 02 08). *Dampak PPKM Berlanjut, Masyarakat Kian Pesimis Lapangan Kerja dan Penghasilan Pulih*. Retrieved 03 17, 2021, from Bisnis.com: https://ekonomi.bisnis.com/read/20210208/9/1353653/dampak-ppkm-berlanjut-masyarakat-kian-pesimis-lapangan-kerja-dan-penghasilan-pulih

Farisa, F. C. (2021, 01 27). *1 Juta Kasus Covid-19 dan Respons Pemerintah...* Retrieved 03 17, 2021, from KOMPAS.com: https://nasional.kompas.com/read/2021/01/27/06405851/1-juta-kasus-covid-19-dan-respons-pemerintah?page=all

Farisa, F. C. (2021, 01 27). *Jokowi: Kesehatan dan Ekonomi Sama Penting, Harus Diselesaikan Bersamaan*. (www.kompas.com) Retrieved 03 15, 2021, from KOMPAS.com: https://nasional.kompas.com/read/2021/01/27/13050811/jokowi-kesehatan-dan-ekonomi-sama-penting-harus-diselesaikan-bersamaan

Kemp, S. (2021, February 11). *Digital 2021: Indonesia*. Retrieved Maret 03, 2021, from DATAREPORTAL: https://datareportal.com/reports/digital-2021-indonesia?rq=indonesia

Makdori, Y. (2021, 03 23). *Survei SMRC: Masyarakat Terbelah soal PPKM Mikro, Ada yang Pro dan Kontra*. Retrieved 04 20, 2021, from Liputan6.com: https://www.liputan6.com/news/read/4513857/survei-smrc-masyarakat-terbelah-soal-ppkm-mikro-ada-yang-pro-dan-kontra

Nasution, M. R., & Hayaty, M. (2019). Perbandingan Akurasi dan Waktu Proses Algoritma K-NN dan SVM dalam Analisis Sentimen Twitter. *Jurnal Informatika*, 226-235.