BAB 2
TINJAUAN PUSTAKA

Pada bab ini akan menjelaskan terkait tinjauan pustaka yang digunakan pada penelitian.

2.1 Analisis Sentimen

Analisis sentimen merupakan sebuah bidang studi yang berfokus dalam menganalisa pendapat, sentimen, sikap, emosi sekaligus penilaian seorang individu terhadap sebuah barang sampai ke sebuah peristiwa yang terjadi. Analisis sentimen sendiri juga memiliki beberapa ranah yang berada langsung di bawahnya seperti penambangan opini (*opinion mining*) dan ekstraksi opini (*extraction opinion*). Hal yang sebelumnya telah disebutkan semuanya masuk kedalam ranah bidang studi analisis sentimen atau penambangan opini. Dalam dunia pendidikan sendiri kata yang lebih sering digunakan adalah "*opinion mining*" sedangkan untuk dunia industri kata yang lebih sering digunakan adalah "analisa sentimen" (Dwi Harijiatno, 2019).

Tugas dasar dari analisis sentimen sendiri adalah untuk dapat mengelompokkan polaritas dari teks-teks yang digunakan sebagai data pada sebuah dokumen, kalimat ataupun fitur. Selanjutnya pendapat yang ada pada dokumen, kalimat ataupun fitur tersebut akan dikelompokkan berdasarkan sifatnya, apakah pendapat yang dikemukakan bersifat positif, negatif atau netral. Jadi sentimen analisis berfokus pada pengolahan pendapat atau opini yang didalamnya mengandung polaritas yang memiliki nilai sentimen baik nilai sentimen tersebut positif, negatif maupun netral (Ardiani dkk., 2020).

Sentimen analisis sendiri dibagi menjadi 2 kelompok berdasarkan darimana sumber datanya berasal. Kelompok pertama adalah kelompok *coarse-grained sentiment analysis* yang berfokus pada level dokumen-dokumen. Analisis sentimen ini secara garis besar difokuskan untuk menganggap seluruh isi sebuah dokumen sebagai sebuah sentimen yang memiliki sifat atau polaritas. Kelompok kedua adalah kelompok *fined-grained sentiment analysis* yang dimana kelompok analisis sentimen ini berfokus pada level kalimat. Fokus utama dari kelompok analisis sentimen ini adalah untuk menentukan sentimen pada setiap kalimat-

kalimat yang ada. Analisis sentimen ini dapat diklasifikasikan pada beberapa kelas sentimen yaitu kelas sentimen yang memiliki sifat positif, negatif dan netral.

2.2 *Lexicon*

Dalam melakukan pelabelan ada beberapa metode yang dapat diimplementasikan. Salah satu metode pelabelan tersebut adalah metode *lexicon*. Saat melakukan suatu analisa, unit-unit pada dokumen atau teks harus memiliki bentuk dan makna. Disini lexicon akan berperan dalam menentukan bentuk dan makna dari dokumen atau teks tersebut. Lexicon akan melabeli sebuah dokumen atau teks berdasarkan kamus yang sebelumnya dibuat atau digunakan. Lexicon tidak hanya menentukan bentuk dan maknanya akan tetapi juga menentukan aturan kombinasi untuk menentukan bagaimana makna dari teks tersebut dihitung (Cruse & Fields, 2017).

Dalam melakukan pelabelalan lexicon sendiri, akan digunakan 2 kamus untuk memboboti polaritas sebuah kata. Kamus tersebut adalah kamus positif dan kamus negatif. Kamus ini yang sebelumnya telah dibuat untuk menjadi acuan seberapa besar bobot sebuah kata dalam suatu kalimat, baik bobot positifnya maupun bobot negatifnya. *Lexicon* sendiri akan menentukan polaritas sebuah kaliamat setelah menjumlahkan bobot positif dan bobot negatif setiap kata yang ada pada kalimat tersebut.

2.3 *Text Classification*

Text classification atau klasifikasi teks merupakan sebuah metode atau pekerjaan untuk dapat menentukan jika sebuah dokumen merupakan salah satu dari kategori yang sebelumnya telah ditentukan. Klasifikasi teks sendiri termasuk dalam *supervised learning* yang dimana ada empat tahapan didalamnya. Empat tahap tersebut adalah *preprocessing,* rekayasa fitur, generasi model klasifikasi dan terakhir adalah pengkategorian dokumen. Pada tahap pertama *preprocessing*, dokumen diubah menjadi bentuk vektor yang berarti teks yang ada pada dokumen tersebut harus dipisah menjadi kata-kata yang terpisah. Pada tahap ini juga dilakukan pembersihan terhadap *stopword* untuk menghapus kata-kata yang bersifat umum dan tidak memiliki makna pada dokumen tersebut sesuai dengan kosakata yang digunakan sebelumnya. Setelah *stopword* yang ada pada teks dokumen tersebut dibersihkan maka dilakukanlah *stemming* agar dapat mencari

kata dasar yang telah diekstrak dari dokumen tersebut. Langkah selanjutnya adalah tahap rekayasa fitur yang merupakan tahap *training* atau tahap pelatihan yang terdiri dari beberapa tahapan seperti tahapan seleksi untuk fitur, *feature weighting* dan pembangunan kamus (*dictionary construction*). Tahap tersebut bertujuan untuk menghapus semua fitur-fitur tidak relevan yang selalu ada pada dokumen. Langkah ketiga adalah melakukan generalisasi model klasifikasi yang merupakan tahapan untuk membangun algoritma klasifikasi yang ingin digunakan ataupun diimplementasikan. Tahapan terakhir adalah pengkategorian dokumen yang bertujuan untuk dapat melakukan klasifikasi terhadap dokumen baru yang sebelumnya didapatkan. Pada tahap ini dokumen harus telah melewati tahapan yang sebelumnya dijelaskan yaitu tahapan *preprocessing* dan tahapan *feature weighting* (Suharno dkk., 2017).

2.4 *Machine Learning*

Machine learning merupakan sebuah pendekatan dalam bidang *Artificial Intelligence* (AI) atau yang lebih dikenal sebagai kecerdasan buatan. Tujuan dari *machine learning* sendiri adalah untuk menggantikan sekaligus menirukan perilaku-perilaku dari manusia. Tujuan lain dari *machine leaning* adalah untuk dapat menyelesaikan masalah-masalah yang ada ataupun melakukan optimisasi. *Machine learning* sendiri menggunakan fungsi dari komputer untuk dapat memprediksi sampai memahami sifat-sifat ataupun ciri-ciri dari sebuah objek. *Machine learning* mempelajari dan mengidentifikasi pola (*pattern*) dari objek tertentu berdasarkan *dataset* yang diberikan. *Machine learning* memiliki ciri khas yaitu memiliki proses-proses seperti proses pelatihan, proses pembelajaran dan proses *training* untuk dapat menjalankan *Machine learning* dengan baik. Oleh karena hal tersebut *Machine learning* membutuhkan data-data sebagai *dataset* untuk dipelajari sebagai data *training*. Hasil dari proses *training* tersebut yang akan menjadi acuan untuk *Machine learning* melakukan langkah-langkah selanjutnya.

Machine learning dikelompokkan menjadi tiga kelompok utama yaitu *supervised learning* atau pembelajaran terarah, *unsupervised learning* atau pembelajaran tidak terarah dan yang terakhir adalah *reinforcement learning*. *Supervised learning* atau pembelajaran terarah merupakan sebuah pembelajaran

yang dimana output yang diharapkan telah diketahui sebelum dilakukannya pembelajaran. Dengan kata lain pembelajaran ini menggunakan data yang sebelumnya sudah tersedia. Contoh dari *supervised learning* sendiri adalah *text classification*, *spam detection*, *object detection* dan lain-lain. *Unsupervised learning* atau pembelajaran tidak terarah merupakan pembelajaran yang dimana tidak memerlukan target output. Pada metode pembelajaran *Unsupervised learning,* hasil tidak dapat ditentukan seperti yang diharapkan selama proses pembelajaran berlangsung. Nilai bobot yang disusun pada proses untuk *range* tersebut tergantung pada nilai output yang diberikan sebelumnya. Contoh dari *unsupervised learning* adalah *anomaly detection*, *noise removal* dari *dataset* dan lain-lain. Terakhir adalah *reinforcement learning* yang bertujuan untuk dapat menggunakan pengamatan sekaligus pengumpulan data melalui interaksi langsung untuk menentukan tindakan selanjutnya. Contoh dari *reinforcement learning* adalah *self-driving cars*, *ad-recommendation system* dan lain-lain (WESLEY, 2018).

2.5 *Naïve Bayes* dan *Gaussian Naïve Bayes Classification*

Naïve Bayes adalah suatu metode pengklasifikasian yang mengaplikasikan Teorema Bayes dengan probabilitas sederhana dan dengan tingkat asumsi independen (ketidaktergantungan) yang tinggi. *Naïve bayes* sendiri menggunakan pengalaman pada masa lalu untuk dapat memprediksi peluang pada masa depan sehingga dikenal dengan nama Teorema Bayes. Keuntungan dari menggunakan metode ini adalah metode ini hanya memerlukan data pelatihan dalam jumlah kecil agar metode ini dapat melakukan penentuan estimasi terhadap parameter yang diperlukan. *Naïve bayes* juga sering bekerja dengan performa yang lebih baik jika dihadapkan pada situasi pada dunia nyata yang memiliki tingkat kompleksitas yang lebih tinggi dari pada yang sebelumnya diharapkan (Manalu dkk., 2017). *Naïve bayes* memiliki rumus secara umum yang ditunjukkan pada Persamaan (2.1).

$$P(\ \ |x) = \frac{P\ (h|x)P(h)}{P(x)} \tag{2.1}$$

Keterangan rumus :

x : Merupakan data yang kelasnya belum diketahui

h : Hipotesis untuk data *x* yang merupakan sebuah kelas spesifik

$P(\ |x)$: Probabilitas untuk hipotesis *h* yang didasarkan pada kondisi *x* (posterior prob).

$P(\ )$: Probabilitas untuk hipotesis *h* (prior prob).

$P(x)$: Probabilitas untuk *x*.

Yang dimana aturan bayes tersebut dapat diartikan jika suatu dokumen *x* memiliki nilai akhir berupa $P(\ 1|x) < P(\ 2|x)$. Maka dokumen *x* dapat diklasifikasikan sebagai bagian dari kelas *h2*. Pernyatan $P(\ 1|x)$ sendiri menunjukkan bahwa probabilitas pada *h1* didasarkan pada kondisi *x*, sama halnya dengan *h2*.

Sedangkan untuk algoritma *Naïve Bayes Classifier* sendiri, untuk setiap dokumen akan dipresentasikan dengan pasangan atribut $(x_1x_2x_3x_4 \dots x_n)$. Yang dimana x_1 merepresentasikan kata pertama sampai ke x_n yang merepresentasikan kata *n*. v merupakan himpunan kategori pada data. Pada saat dilakukannya klasifikasi, algoritma akan mencari probabilitas tertinggi pada semua kategori yang direpresentasikan oleh (v_{map}). Persamaan dari algoritma *Naïve Bayes Classifier* sendiri dapat dilihat pada Persamaan (2.2).

$$\hat{v} = \underset{v}{argmax}\left(\frac{P(x_1x_2x_3x_4 \dots x_n|Pv_j)}{P(x_1x_2x_3x_4 \dots x_n)}\right) \quad (2.2)$$

Karena $P(x_1x_2x_3x_4 \dots x_n)$ bernilai konstan pada semua kategori (v_j) maka Persamaan (2.2) dapat ditulis ulang menjadi Persamaan (2.3).

$$\hat{v} = \underset{v}{argmax}\left(P(x_1x_2x_3x_4 \dots x_n|v_j)P(v_j)\right) \quad (2.3)$$

Persamaan diatas lalu dapat disederhanakan menjadi Persamaan yang lebih sederhana seperti yang ditunjukkan oleh Persamaan (2.4).

$$\hat{v} = \underset{v}{argmax} \prod_{i=1}^{n} P(x_i|v_j)P(v_j) \quad (2.4)$$

Yang dimana :

v = Himpunan kategori dari data

$\hat{v}$ = Semua kategori yang diuji.

v_j = Kategori data j = 1, 2, 3, 4, …. n.

$P(x_i|v_j)$ = Probabilitas x_i berdasarkan probabilitas v_j.

$P(v_j)$ = Probabilitas dari v_j.

$\underset{v}{argmax}$ = Nilai maksimum dari himpunan kategori data.

Untuk mendapatkan $P(v_j)$ dan untuk mendapatkan $P(x_i|v_j)$ maka akan dilakukan perhitungan pada saat pelatihan atau *training* yang dimana persaman dari keduanya ditunjukkan oleh Persamaan (2.5) dan Persamaan (2.6).

$$P(v_j) = \frac{|dosc\ j|}{|contoh|} \quad (2.5)$$

$$P(x_i|v_j) = \frac{n_k+1}{n+|kosakata|} \quad (2.6)$$

Yang dimana :

$|dosc\ j|$ = Jumlah dokumen untuk setiap kategori j.

$|conto\ |$ = Jumlah dokumen untuk semua kategori.

n_k = Jumlah frekuensi untuk kemunculan tiap kata.

n = Jumlah frekuensi kemunculan kata pada setiap kategori.

$|kosakata|$ = Jumlah semua kata dari semua kategori yang ada.

Dapat dilihat untuk Persamaan (2.6) terdapat penambahan 1 pada pembilang, hal ini dilakukan agar dapat mengantisipasi apabila ada suatu kata yang terdapat pada dokumen data *testing* yang tidak terdapat pada setiap dokumen *training*.

Salah satu tipe dari implementasi spesial *Naïve Bayes* sendiri merupakan *Gaussian Naïve Bayes*. *Gaussian Naïve Bayes* sendiri merupakan tipe *Naïve Bayes* yang digunakan ketika nilai bersifat *continuous* (Agarwal dkk., 2019). Pada *Gaussian Naïve Bayes* diasumsikan bahwa setipa nilai yang *continuous* didistribusikan berdasarkan distribusi normal atau distribusi Gauss. Adapun Persamaan dari *likelihood Gaussian Naïve Bayes* diperlihatkan pada Persamaan (2.7).

$$p(x_i|y) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(\frac{(x_i-\mu_y)^2}{2\sigma_y^2} \right) \quad (2.7)$$

Keterangan rumus :

π = Nilai Phi 3.142

σ = Variance Dari Data

μ = Mean Dari Kelas y

x_i = Nilai Dari Kelas-i

Untuk mendapatkan nilai dari μ (Mean) maka digunakan Persamaan (2.8) yang diperlihatkan di bawah ini :

$$\mu = \Sigma \frac{x_i}{n} \quad (2.8)$$

Keterangan rumus :

μ = Mean

x_i = Merupakan nilai data ke-i

n = Banyaknya data

Setelah mendapatkan Mean dari Persamaan (2.8) maka langkah selanjutnya adalah untuk menemukan *variance* (σ) dari setiap kelas. Untuk Persamaan dari *variance* diperlihatkan pada Persamaan (2.9).

$$\sigma = \frac{\Sigma(x_1 - \mu)^2}{n-1} \quad (2.9)$$

Keterangan rumus :

x_i = Merupakan Dokumen ke-i

$\bar{x}$ = Merupakan nilai mean

n = Banyaknya data

μ = Mean

σ = *Variace*

Setelah mendapatkan Mean dan *Variance* dari kedua kelas maka digunakan Persamaan (2.7) untuk mendapakan nilai model dari sebuah dokumen. Setelah nilai model ditemukan maka digunakan Persamaan probabilitas yang ditunjukkan pada Persamaan (2.4) untuk menghitung masing-masing nilai dari setiap kelas agar dapat menentukan sentimen dari dokumen tersebut.

2.6 *Performance Evaluation Measure*

Performance Evaluation Measure atau yang lebih dikenal dengan nama pengukuran evaluasi performa merupakan suatu tahapan yang digunakan agar dapat mengukur performa suatu sistem yang sedang digunakan. Pengukuran evaluasi performa dalam berbagai kasus sering digunakan pada data training agar dapat digunakan untuk mengevaluasi model yang sudah dibangun. Pengukuran evaluasi performa memiliki beberapa perhitungan yang dapat digunakan sebagai kombinasi maupun secara parsial, beberapa perhitungan dari *Performance*

Evaluation Measure dapat dilihat dari Persamaan (2.11) untuk *precission*, Persamaan (2.12) untuk *accuration*, Persamaan (2.13) untuk *recall* dan persamaan (2.14) untuk *f1-measures*.

$$Precision = \frac{TP}{FP+TP} \quad (2.11)$$

$$Accuration = \frac{TN+TP}{FN+FP+TN+TP} \quad (2.12)$$

$$Recall = \frac{TP}{FN+T\ P} \quad (2.13)$$

$$F1\quad Score = \frac{2\ x\ Precision\ x\ Recall}{Precision\ x\ Recall} \quad (2.14)$$

$$F1\quad Score(weig\ ted) = \frac{(E_{c1}\times|c1|)+(E_{c2}\times|c2|)}{|c1|+|c2|} \quad (2.15)$$

Precision merupakan sebuah tingkat ketepatan yang dihitung dengan membandingkan antara prediksi positif dibagi dengan total hasil yang diprediksi positif. *Accuration* merupakan perbandingan pada informasi yang dihasilkan oleh sistem yang bersifat benar dengan keseluruhan informasi yang tersedia. Sedangkan recall merupakan seberapa tepat informasi yang sama dengan informasi yang sebelumnya telah dipanggil. Selanjutnya adalah *F1-Score, F1-score* sendiri merupakan nilai tengah dari hasil *precision* dan hasil *recall*. Terakhir adalah *F1-score weighted*, *1-score weigthed* sendiri merupakan nilai tengah dari *precission* dan *recall* yang dimana dihitung berdasarkan mean dari semua kelas yang ada. *F1-score* sendiri akan mempertimbangkan nilai *support* (jumlah data kelas pada evaluasi) setiap kelas data saat melakukan perhitungannya.

Pengukuran evaluasi performa juga biasanya digambarkan sebagai sebuah Tabel yang didalamnya berisi *dataset* yang berisi kelas-kelas *true* dan *false*. Tabel tersebut bernama table *confusion matrix* yang dapat dilihat seperti Tabel 2. 1 :

Tabel 2. 1 Confusion Matrix

Class	*Predicted Class*	
True Class	*Positive*	*Negative*
Positive	TP	FN
Negative	FP	TN

Pada Tabel 2.1 TP merupakan *true positive* yang menunjukkan jumlah untuk prediksi kelas yang diprediksi memiliki nilai positif sekaligus kelas aktualnya bernilai positif. FN merupakan *false negative* yang menunjukkan jumlah kelas yang diprediksi memiliki nilai negatif sedangkan kelas aktualnya bernilai positif. Selanjutnya FP sendiri merupakan *false positive* yang menunjukkan jumlah kelas yang diprediksi memiliki nilai positif sedangkan untuk kelas aktualnya bernilai negatif. Terakhir adalah TN yang merupakan *true negative* yang menunjukkan jumlah kelas yang diprediksi bernilai negatif sekaligus kelas aktualnya bernilai negatif juga.

2.7 *Split Validation*

Split validation merupakan sebuah teknik validasi yang membagi data menjadi dua tipe tipe secara acak. *Split validation* membagi sebagian data menjadi data training dan sebagian menjadi data testing. Pada split validation sendiri terdapat dua konsep yaitu konsep *training error* dan konsep *test error*. konsep *training error* didapatkan dengan cara melakukan perhitungan terhadap klasifikasi model yang sama dengan model yang digunakan untuk pelatihan. Sedangkan untuk konsep *test error* didapatkan dengan cara menggunakan dua buah data yang terpisah sepenuhnya. Dua data yang ada berupa data untuk melatih mode atau data latih dan yang satu lagi adalah data untuk menghitung kesalahan uji atau data uji. Kedua *dataset* pada *split validation* ini harus memiliki nilai-nilai label yang sama.

Pada *split validation*, data testing tidak digunkaan saat melakukan pelatihan model. Sehingga model tidak mengetahui hasil atau nilai asli dari data tersebut. Hal tersebut dilakukan untuk dapat mengetahui nilai akurasi dari model yang dikembangkan. Jika model memiliki akurasi yang tinggi maka model tersebut dapat dikatakan sebagai model yang baik.

2.8 *Text Mining*

Text mining atau yang dikenal juga dengan *intelligent Text Analysis* atau *text data mining* secara umum merupakan sebuah proses dalam mengekstraksi pola yang menarik dan sepele atau pengetahuan-pengetahuan dari teks yang berada pada dokumen yang tidak terstruktur. *Text mining* juga dikarakteristikkan sebagai proses untuk menganalisa text untuk dapat mengekstraksi informasi-

informasi yang nantinya dapat berguna untuk tujuan yang tertentu. Jika dibandingkan dengan data yang disimpan pada *database*, text yang digunakan tidak berstruktur, amorf, dan sulit untuk ditangani secara algoritmik. Namun demikian pada saat ini teks merupakan sebuah kendaraan yang paling sering digunakan untuk secara formal bertukar informasi satu sama lain (Khan dkk., 2020).

Dalam melakukan *text mining*, data sebelumnya harus dipersiapkan terlebih dahulu. Proses dalam melakukan persiapan data ini dikenal sebagai proses *preprocessing* data. Proses ini sendiri dilakukan agar data yang sebelumnya belum teratur atau belum terstruktur dapat menjadi lebih terstruktur dan teratur agar data tersebut dapat digunakan atau diolah. Adapun tahapan dalam *preprocessing* dapat dilihat pada sub sub Bab 2.8.1 sampai sub sub Bab 2.8.6.

2.8.1 *Spelling Normalization*

Langkah ini merupakan langkah untuk memperbaiki kata singkatan ataupun ejaan-ejaan yang salah. Awalnya mesin akan mengecek jika dalam suatu kalimat terdapat kata yang tidak baku. Jika benar terdapat kalimat tidak baku, maka mesin akan memperbaiki kata tersebut berdasarkan kamus yang digunakan. Jika semua kata dalam setiap kalimat sudah baku dan semua kata tidak baku telah diperbaiki, maka data akan disimpan yang kemudian akan digunakan untuk tahap selanjutnya.

Spelling Normalization ini dilakukan agar tidak terjadi perluasan dalam jumlah dimensi kata yang dapat mengganggu sampai merusak proses penyusunan matriks (WESLEY, 2018). Langkah awal dalam spelling *normalization* adalah data dimasukkan. Setelah data dimasukkan selanjutnya data akan dicek menggunakan kamus untuk melihat apakah ada kata yang tidak baku yang bisa diperbaiki. Jika ada kata yang tidak baku maka akan diperbaiki, jika tidak ada maka data akan disimpan. Setelah selesai dengan tahap *normalizing* maka data akan disimpan dan digunakan untuk tahap selanjutnya

2.8.2 *Case Folding*

Case Folding merupakan tahap dimana dilakukan standarisasi huruf kapital dalam sebuah kalimat menjadi huruf kecil atau *lowercase*. Pada tahap ini sistem akan mengecek apakah di dalam data terdapat huruf kapital atau tidak. Jika

tidak ada huruf kapital yang berada di dalam kalimat maka akan langsung disimpan. Sedangkan jika terdapat huruf kapital dalam kata maka sistem akan mengubahnya terlebih dahulu menjadi huruf kecil atau *lowercase* yang kemudian akan dihasilkan data hasil dari dilakukannya tahap *case folding.*

Tujuan utama dilakukannya tahap ini agar dapat menghindari terjadinya perulangan data yang berbeda pada huruf kapital saja (Halimi & Rudyanto Arief, 2021). Langkah pertama pada tahap ini adalah data yang akan digunakan dimasukkan. Setelah itu data akan dicek jika didalamnya terdapat kata yang memiliki huruf kapital. Jika ada maka huruf tersebut akan diubah menjadi huruf kecil, jika tidak ada maka data akan disimpan. Setelah selesai dengan tahap *case folding* maka data akan disimpan dan digunakan untuk tahap selanjutnya.

2.8.3 *Filtering Stopword*

Filtering Stopword merupakan sebuah cara untuk menyeleksi kata penting yang berada di dalam kalimat. Didalam kalimat biasanya terdapat sebuah kata yang kurang bermakna atau kata yang memiliki nilai informasi rendah. Kata-kata ini yang biasanya disebut sebagai Stopword. *Filtering Stopword* ini dilakukan untuk menghilangkan kata-kata yang kurang penting atau tidak bermakna tersebut berdasarkan kamus yang digunakan (Halimi & Rudyanto Arief, 2021).

Filtering Stopword ini penting dalam proses klasfikasi suatu dataset. Filtering Stopword bisa mengurangi ukuran dari suatu dataset yang digunakan dalam klasifikasi. Hal tersebut nantinya akan mengurangi waktu dari training akibat jumlah token dalam kamus lebih sedikit. Hasil dari tahap *filtering stopword* ini nantinya akan disimpan dan digunakan untuk tahap selanjutnya

2.8.4 *Tokenizing*

Tokenizing atau Tokenisasi merupakan sebuah langkah untuk memecah kalimat menjadi per kata-kata berdasarkan kata penyusun kalimat tersebut. Langkah pertama pada *Tokenizing* sendiri adalah memasukkan data hasil *normalizing*. Kemudian setelah data tersebut dimasukkan maka akan dilakukan pemecahan dokumen dengan memotong string berdasarkan kata-kata yang ada dalam dokumen. Setelah dilakukan pemotongan kata maka dokumen disimpan sebagai data hasil *tokenizing*.

Pada langkah ini semua kata dalam kalimat akan dipecah menjadi *string* (WESLEY, 2018). Langkah awal dari tahap ini adalah, data yang awalnya akan digunakan dimasukkan. Setelah itu setiap kata dalam kalimat akan dipecah menjadi string. Setelah semua kata pada data dipecah menjadi string kemudian data akan disimpan untuk digunakan pada tahap selanjutnya. Tahap *tokenizing* akan berguna nantinya dalam proses ekstraksi fitur karena proses ini akan membantu untuk menilai setiap kata yang ada dalam kalimat.

2.8.5 *Stemming*

Stemming merupakan cara untuk mengurangi pembentukan kata baru (infleksi) akibat adanya penambahan imbuhan (sufiks dan afiks). Sebagai contoh kata "memakan" akan diubah menjadi kalimat akarnya yaitu "makan" para proses ini. Pada proses *stemming* berbahasa Indonesia memiliki proses yang berbeda dari *stemming* berbahasa Inggris. Pada *stemming* bahasa Indonesia tidak hanya menghilangkan sufiks akan tetapi juga menghilangkan sufiks, prfiks dan konfiks. Tujuan dari *stemming* ini adalah untuk menghasilkan *stem* atau kata yang tersisa ketika menghilangkan imbuhan (sufiks dan afiks) (WESLEY, 2018)..

Langkah pertama dari tahap ini adalah memasukkan data yang sebelumnya akan digunakan. Selanjutnya data tersebut akan dicek jika didalamnya ada kata yang memiliki imbuhan seperti sufiks, infiks, konfiks maupun afiks. Jika ada maka imbuhan tersebut dihapus dan kata akan diubah menjadi kata dasarnya. Setelah data dilakukan *stemming* maka data akan disimpan dan digunakan untuk tahap selanjutnya.

2.8.6 *Cleaning*

Pada sebuah data biasanya sering terdapat simbol atau tanda baca yang tidak diperlukan. Tidak hanya simbol atau tanda baca akan tetapi terdapat juga kata yang tidak baku dan ejaan-ejaan yang tidak tepat yang membuat data menjadi kurang efektif dan tidak memiliki arti/makna. Oleh karena itu dibutuhkan sebuah tahap yaitu tahap *Cleaning*. *Cleaning* sendiri merupakan sebuah tahap untuk menghapus atribut yang tidak berguna seperti link http dan simbol-simbol seperti tanda seru (!)..

Pada *Cleaning*, tahap pertama yang dilakukan adalah data yang akan digunakan dimasukkan. Kemudian data akan dibersihkan melalui beberapa proses

seperti nilai yang hilang diisi, *noise* pada data dihaluskan serta menyelesaikan kata-kata yang tidak konsisten. Setelah data telah dibersihkan maka data akan disimpan dan digunakan untuk tahap selanjutnya. Tahap *cleaning* sendiri memiliki tujuan untuk mengurangi beban dalam melakukan proses *training*(WESLEY, 2018)

2.9 Ekstraksi Fitur

Ekstraksi fitur merupakan sebuah proses untuk mengubah sebuah data mentah menjadi numerik yang dapat diproses sambil mempertahankan informasi-informasi didalamnya. Ekstraksi fitur biasanya menghasilkan hasil yang lebih baik jika diimplementasikan pada *machine learning* diibandingkan langsung menggunakan data mentah pada *machine learning*. Salah contoh ekstraksi fitur adalah *Bag Of Word* atau yang disingkat sebagai BOW. Ekstraksi fitur ini sangat populer dan mempunyai performa yang cukup bagus saat memilih dan mengklasifikasikan fitur dengan membuat kantong (*bag)* untuk setiap jenis-jenis instansi atau kata. Ekstraksi fitur BOW sendiri mengimplementasikan penggunaan *Natural Language Processing* dan *Information Retrival* secara sederhana. Pada ekstraksi fitur BOW, kata duplikat, tata bahasa dan urutan kata tidak diperhatikan (Qader dkk., 2019).

Contoh lain dari ekstraksi fitur adalah ekstraksi fitur *TF-IDF* atau *Term Frequency-Inverse Document Frequency* yang merupakan pengembangan dari ekstraksi fitur *Bag Of Word*. *Term frequency* (*TF*) pada *TF-IDF* mengacu pada semakin tinggi sebuah frekuensi dari term maka pada suatu dokumen, maka nilai bobot untuk *term* tersebut akan akan juga semakin tinggi. Sedangkan untuk *Inverse Document Frequency* (*IDF*) pada *TF-IDF* mengacu pada kebalikan dari proses *Term frequency* (*TF*). Pada *Inverse Document Frequency* (*IDF*) semakin tinggi kemunculan dari sebuah *term* maka semakin rendah bobot dari *term* tersebut (Zakiyuddin, 2021).

TF-IDF sendiri memiliki sebuah Persamaan yang ditunjukkan pada Persamaan (2.16) dan (2.17).

$$Bobot \quad : W_{i,j} = tf\ x\ idf \tag{2.16}$$

$$: W_{ij} = tf_{i,j}\ x\ \log\frac{N}{n} \tag{2.17}$$

dimana :

$W_{i,j}$ = bobot dokumen ke i, terhadap kata ke j

$tf_{i,j}$ = banyaknya kata i yang muncul pada dokumen j

N = Jumlah semua dokumen dalam kumpulan dokumen

N = Jumlah dokumen yang didalamnya terdapat kata t_j

2.10 Twitter dan Twitter API

Twitter (dibaca "/ tw t r/") merupakan sebuah layanan jejaring sosial sekaligus sebuah mikroblograring yang mengizinkan para penggunanya untuk dapat mengirimkan sekaligus membaca sebuah pesan berbasis teks. Pesan tersebut dibatasi sampai 140 karakter panjangnya yang lebih dikenal oleng penggunanya dengan nama kicauan atau *tweet. Twitter* sendiri berdiri pada awal tahun 2006 tepatnya tanggal 21 maret 2006 di San Francisco, California, Amerika serikat oleh Jack Dorsey dan diluncurkan pada bulan Juli 2006. Sejak diluncurkannya pada bulan Juli 2006, *Twitter* langsung melesat menjadi salah satu dari sepuluh besar situs pada Internet yang paling sering dikunjungi oleh para pengguna Internet itu sendiri dan dijuluki sebagai "pesan singkat dari Internet". Pada *Twitter*, para penggunanya tidak perlu untuk melakukan pendaftaran untuk dapat membaca kicauan yang ada. Akan tetapi hanya para pengguna yang terdaftar pada *Twitter* yang bisa menulis kicauan atau *tweet* lewat situs antarmuka web, SMS atau pesan singkat maupun melalui berbagai aplikasi perangkat seluler (Zukhrufillah, 2018)

Twitter juga memiliki API *Twitter*, API *Twitter* atau yang dikenal dengan nama *Application Programming Interface Twitter* merupakan sebuah program yang disediakan langsung oleh *Twitter* agar para pengembang (*developer*).*Twitter API* sendiri dikembangkan untuk mempermudah pengaksesan informasi yang ada pada *Twitter*. Adapun para pengguna yang ingin menjadi menggunakan sistem ini harus mendaftar terlebih dahulu pada laman yang disediakan oleh *Twitter*. Setelah mereka terdaftar, maka para pengembang tersebut akan mendapatkan 3 *key* yang nantinya akan berguna sebagai salah satu syarat autentikasi dari aplikasi. Tujuan dari autentikasi tersebut adalah agar para pengembang dapat mengunduh data-data yang mereka perlukan pada *Twitter*.

2.11 Penelitian Terdahulu

Berdasarkan literatur yang telah digunakan pada penulisan tugas akhir ini, dapat disimpulkan bahwa dengan dilakukannya analisis sentimen terhadap Covid-19 varian Omicron ini memberikan kemudahan bagi masyarakat untuk mengetahui penilain masyarakat terhadap Covid-19 varian Omicron. Adapun data yang digunakan sebagai *dataset* penelitian ini diambil dari platform media sosial *Twitter* dengan menggunakan bantuan *Twitter API.* Data yang diambil dari *Twitter API* tersebut akan berbentuk *tweet* yang berisi opini-opini masyarakat dengan kata kunci Covid-19 Omicron.

Untuk melakukan penelitian ini diperlukan metode yang tepat dan sesuai dengan kebutuhan Penulis . Adapun metode yang digunakan pada penelitian ini adalah metode *Gaussian Naïve Bayes Classifier*. Kelebihan metode ini adalah hanya memerlukan data pelatihan (*Training Data*) jumlah kecil agar metode ini dapat melakukan penentuan estimasi terhadap parameter yang diperlukan dalam proses klasifikasi. Metode *Gaussian Naïve Bayes Classifier* sendiri akan digunakan untuk menentukan opini masyarakat terhadap Covid-19 varian Omicron. Selanjutnya akan dilakukan pengelompokan data berdasarkan sifatnya, apakah data tersebut condong kearah positif, negatif atau netral.

Dalam menjalankan penelitian ini, Penulis menggunakan beberapa penelitian yang telah dilakukan sebelumnya sebagai acuan daripada penelitian ini. acuan yang digunakan dalam penelitian ini dapat dilihat pada Tabel 2. 2 :

Tabel 2. 2 Penelitian Terdahulu

No	Nama Penulis dan Tahun Publikasi	Hasil
1	Ardiani dkk., 2020	**Metode** : *Naïve Bayes.* **Hasil** : Nilai akurasi 72%, presisi 72% dan *recall* sebesar 72%.
2	Arsi dkk., 2021	**Metode** : *Support Vector Machine* **Hasil** : Analisa Mendapat akurasi sebesar 81.15% yang sebelumnya hasil yang didapatkan sebesar 79.06%.
3	Chakraborty dkk., 2020	**Metode** : *Deep Learning.* **Hasil** : Analisa Sentimen pada *Twitter* mendapatkan akurasi sebesar

		81%.
4	Fauziyyah, 2020	**Metode** : *Text Blob Polarity.* **Hasil** : Analisa menghasilkan kelas tertinggi yaitu netral dengan polaritas 58.97%
5	Halimi & Rudyanto Arief, 2021	**Metode** : *Lexicon* dan *K-Nearest Neighbor.* **Hasil** : Akurasi pada metode Lexicon pada K3 sebesar 80.92% sedangkan K-Nearest Neighbor pada K3 sebesar 80.66% yang membuat metode Lexicon pada K3 lebih baik dibandingkan K-Nearest Neighbor.
6	Imron, 2019	**Metode** : *Naïve Bayes.* **Hasil** : Analisa sentimen menghasilkan akurasi sebesar 85.8%.
7	Ruhyana, 2019	**Metode** : *Naïve Bayes.* **Hasil** : Evaluasi performa pada model Menghasilkan akurasi sebesar 86.67%, presisi sebesar 71.43% dan recall sebesar 80.00%
8	WESLEY, 2018	**Metode** : *Naïve Bayes, Support Vector Machine, Logistic Regression, Decision Tree,* dan *K-Nearest Neighbor.* **Hasil** : Dari kelima algoritma akurasi tertinggi dicapai oleh *Support Vector Machine* dengan akurasi sebesar 95.6%
9	Yulita dkk., 2021	**Metode** : *Naïve Bayes.* **Hasil** : Analisa sentimen menghasilkan akurasi sebesar 93%.