BAB II

TINJAUAN PUSTAKA

II.1. Cyberbullying

Cyberbullying (perundungan dunia maya) adalah *bullying*/perundungan menggunakan menggunakan teknologi digital sebagai medianya. *Cyberbullying* biasanya terjadi di jejaring sosial, *platform chatting*, *platform game*, dan *smartphone*. *Cyberbullying* termasuk tindakan berulang dengan tujuan menakuti, membuat marah, atau membuat malu individu atau kelompok yang menjadi sasaran (Livingstone et al., 2020). Contoh:

1. Menyebarkan hal yang tidak benar mengenai individu atau menyebarkan hal memalukan tentang individu atau kelompok di jejaring sosial
2. Mengirim teks berupa ancaman yang menyakitkan melalui media sosial, tanggapan-tanggapan menyakitkan pada fitur komentar media sosial, atau mempublikasi sesuatu yang memalukan/menyakitkan
3. Duplikasi atau berpura-pura menjadi seseorang (misalnya menggunakan akun anonim dan tidak benar) kemudian mengirim kata-kata mengandung ujaran kebencian kepada orang lain.

Perundungan (*bully*) secara langsung dan perundungan dunia maya (*cyberbullying*) mungkin saja dapat terjadi secara beriringan. Namun, *cyberbullying* punya jejak digital yang berupa sebuah rekaman atau catatan

yang sangat berguna dan dapat menjadi sebuah bukti sebagai bahan pemeriksaan untuk menghentikan tindakan cyberbullying yang berkepanjangan.

Dampak dampak dari *cyberbullying* dapat bertahan lama dan memengaruhi seseorang dalam banyak cara, meliputi (Livingstone et al., 2020):

1. Secara mental: perasaan kesal, malu, bodoh, bahkan marah.
2. Secara emosional: perasaan malu atau kehilangan minat pada hal-hal yang disukai.
3. Secara fisik: perasaan lelah karena kurang istirahat atau mengalami gejala seperti sakit perut dan sakit kepala

Tindakan *cyberbullying* dapat membuat seseorang tidak ingin membicarakan atau mengatasi masalah tersebut. Bahkan *cyberbullying* dapat mengakibatkan seseorang mengakhiri nyawanya akibat tekanan mental. *Cyberbullying* didefinisikan sebagai tindakan penyalahgunaan sosial media yang diatur oleh Undang-Undang Informasi dan Transaksi Elektronik (UU ITE) No. 11 Tahun 2008 pasal 27 ayat 3 yang berisi tentang pencemaran nama baik ataupun penghinaan”

II.2. Text Mining

Text mining adalah penerapan teori dan teknik dari data mining dalam menyelesaikan masalah pada teks, di mana analisa teks tersebut bertujuan untuk menemukan informasi yang bermanfaat sesuai masalah penelitian (Septian *et al.*, 2017). *Text mining* yaitu teknik yang dipakai dalam menyelesaikan masalah klasifikasi, sentimen analisis, *text clustering*, proses ekstraksi teks dan *Intelligence Information Retrieval* (Sari & Stevenson, 2016).

Text mining merupakan bagian dari teknik data mining itu sendiri. Keduanya bertujuan untuk mendapatkan informasi dan *knowledge* (pengetahuan) yang memiliki arti dari sekumpulan data yang besar dan kompleks. Data tersebut bisa berbentuk sebuah database. Tetapi, letak perbedaannya adalah pada jenis datanya. Data mining menggunakan input data berdasarkan data yang terstruktur sedangkan *text mining* menggunakan data yang tidak terstruktur atau semi terstruktur. Perbedaan ini menyebabkan adanya tantangan pada penyelesaian peneltian pada bidang *text mining*, yaitu kompleksitas struktur teks yang tidak lengkap, arti kurang jelas dan tidak baku, menggunakan bahasa yang berbeda serta penerjemahan yang kurang presisi (Sussolaikah & Alwi, 2016).

Media sosial yang merupakan sumber potensial data terstruktur dianggap sebagai sumber informasi intelijen pasar dan pelanggan yang berharga. Banyak perusahaan yang menggunakan text mining untuk menganalisis atau memprediksi kebutuhan pelanggan dan menilai persepsi merek mereka. Analisis teks dapat mengatasi isu-isu dengan menganalisis volume besar dari data terstruktur, mengeluarkan pendapat, emosi dan sentimen dan hubungan mereka dengan merek dan produk.

II.3. Algoritma Naive Bayes

II.3.1. Konsep Algoritma Naive Bayes

Algoritma Naïve Bayes dikembangkan oleh R.T. Bayes pada abad ke-18. Klasifikasi Naïve Bayes adalah pengklasifikasian berdasarkan ilmu statistik yang

digunakan untuk menghitung probabilitas keanggotaan suatu kelas. Klasifikasi Naïve Bayes menggunakan konsep dasar teorema Bayes yang mampu melakukan klasifikasi mirip seperti teknik *neural network* dan *decision tree*. Klasifikasi Naïve Bayes terbukti memiliki tingkat akurasi dan kecepatan sangat baik saat diterapkan dalam database yang besar (Kusrini & Taufiq, 2009). Klasifikasi menggunakan teknik Naïve Bayes secara umum dilakukan melalui pendekatan probabilitas atau peluang. Algoritma ini memprediksi probabilitas data selanjutnya berdasarkan pengetahuan dan pengalaman data sebelumnya (Sitepu *et al.*, 2020).

Tujuan algoritma Naïve Bayes adalah untuk melakukan klasifikasi data pada *class* tertentu. Hasil proses klasifikasi diukur dengan nilai *predictive accuracy*. Secara umum, proses algoritma Naïve Bayes adalah sebagai berikut:

1. Hitung jumlah *class*/label.
2. Hitung jumlah kasus yang sama dengan *class* yang sama.
3. Kalikan semua variabel *class*.
4. Bandingkan semua hasil variabel.

Secara sederhana, persamaan algoritma Naïve Bayes yang dapat dijadikan acuan untuk menghitung nilai probabilitas dalam pengambilan suatu keputusan sebagai berikut.

$$P(X|H) = \frac{P(H|X) \cdot P(X)}{P(H)} \tag{2.1}$$

di mana:

X : Data pada kelas yang belum diketahui

H : Hipotesis data X merupakan suatu class spesifik

$P(H|X)$: Peluang hipotesis H berdasarkan kondisi X

$P(H)$: Peluang hipotesis H

$P(X|H)$: Peluang X berdasarkan kondisi pada hipotesis H

$P(X)$: Peluang X

Jurafsky D. dan Martin, J. H. (2018) dalam teknik klasifikasi teks menggunakan persamaan Naive Bayes dengan mengklasifikasikan kategori $c \in C$ dari suatu dokumen $d \in D$ di mana $C = \{c_1, c_2, c_3, \dots, c_i\}$ dan $D = \{d_1, d_2, d_3, \dots, d_i\}$. Dokumen tersebut adalah kumpulan kata-kata yang menyusunnya dan tidak memperhatikan urutan kemunculan kata pada dokumen. Sehingga persamaan sebelumnya dapat ditulis: (Jurafsky & Martin, 2018)

$$P(w_k|c) = \frac{n_k + 1}{n + |V|} \tag{2.2}$$

$w_k|c$: jumlah kata w_k dalam dokumen D yang termasuk kelas c

n_k : jumlah term k yang muncul dalam kelas c

n : total term dalam kelas c

$|V|$: total semua term unik

II.3.2. Kelebihan dan Kekurangan Naive Bayes Secara Umum

Ada beberapa kelebihan dan kekurangan dari algoritma Naive Bayes. Kelebihan dari Baive Bayes adalah sebagai berikut (Widianto, 2019).

1. Flekasibel sehingga dapat digunakan pada data kuantitatif atau kualitatif
2. Relatif tidak membutuhkan data yang banyak
3. Tidak sulit untuk merumuskannya dan mudah dipahami
4. Dalam klasifikasi dokumen bisa diatur-atur, dikondisikan dan disesuaikan dengan kebutuhan setiap masalah penelitian
5. Cenderung tidak perlu mengumpulkan data training yang banyak
6. Memiliki kemampuan perhitungan *Missing value* dengan cara diabaikan dalam perhitungan dan tidak mengurangi kemampuan algoritma.
7. Waktu komputasi yang lebih cepat dan efisien
8. Penyusunan *coding* lebih sederhana
9. Dapat digunakan pada masalah klasifikasi biner maupun *multiclass*

Adapun kekurangan menggunakan algoritma Naive Bayes adalah sebagai berikut:

1. Apabila probabilitas kondisionalnya bernilai nol, maka probabilitas prediksi juga akan bernilai nol
2. Asumsi bahwa masing-masing variabel independen membuat berkurangnya akurasi, karena biasanya ada korelasi antara variabel yang satu dengan variabel yang lain
3. Keakuratannya tidak bisa diukur menggunakan satu probabilitas saja. Butuh bukti-bukti lain untuk membuktikannya.
4. Untuk membuat keputusan, diperlukan pengetahuan awal atau pengetahuan mengenai masa sebelumnya. Keberhasilannya sangat bergantung pada

pengetahuan awal tersebut. Banyak celah yang bisa mengurangi efektivitasnya.

II.4. Support Vector Machine (SVM)

II.4.1. Konsep Support Vector Machine (SVM)

Support Vector Machines (SVM) adalah metode pembelajaran mesin berdasarkan sifat data yang terstruktur yang digunakan untuk klasifikasi dan analisa regresi. Algoritma SVM pertama sekali dilakukan oleh Vladimir Vapnik dan saat ini turunan standarnya dianamakan *Soft Margin* (Cortes & Vapnik, 1995). SVM standar menggunakan himpunan data input kemudian memprediksi setiap masukan yang diberikan. Probabilitas masukan yaitu anggota salah satu kelas dari dua kelas yang ada membuat SVM sebagai penggolong nonprobabilistik linier biner. Algoritma klasifikasi SVM kemudian diberi suatu himpunan pelatihan, yang masing-masing didefenisikan sebagai milik salah satu dari dua kategori, algoritma pelatihan SVM kemudian membangun sebuah model untuk memprediksi apakah data baru masuk ke dalam suatu kategori atau tidak (Sitepu *et al.*, 2021).

Konsep SVM yaitu mencari *hyperplane* paling baik untuk pemisah dua buah kelas pada ruang kelas. Pola adalah anggota dari dua buah kelas: +1 dan -1 dan berbagi alternative garis pemisah (*discrimination boundaries*). Margin yaitu jarak antara hyperplane dengan pola terdekat dari masing-masing kelas. Pola yang paling dekat disebut *support vector*. Proses menentukan posisi hyperplane merupakan inti dari proses pembelajaran SVM. Secara sederhana, teknik SVM adalah

menggambarkan data sebagai titik dalam ruang, kemudian dipetakan sehingga kategori pemisalan terpisah dam terbagi oleh celah yang selebar mungkin. Data baru kemudian dipetakan pada ruang sama yang selanjutnya diperkirakan apakah kategori berdasarkan sisi mana dari celah data tersebut berada.

Tujuan teknik SVM bekerja pada prinsip *Structural Risk Minimization (SRM)* adalah menemukan *hyperlane* terbaik yang memisahkan dua buah kelas pada ruang masukan. Pada dasarnya, SVM merupakan klasifikasi linear. Pemisahan SVM yang baik dapat dicapai oleh *hyperplane* yang memiliki jarak terbesar ke titik data latih dan terdekat dari setiap kelas (disebut margin fungsional) karena pada umumnya semakin besar margin maka semakin kecil kesalahan umum dari pemilah. Misalkan, terdapat m data latih $\{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$ adalah sampel data dan $y_1 \in \{1, -1\}$ merupakan sasaran atau kelas data sampling. Misalkan data untuk kedua kelas terpisah, maka akan dihitung fungsi pemisah (*hyperplane*) (Wu & Kumar, 2009).

$$f(x) = xw + b \tag{2.3}$$

di mana w adalah parameter bobot dan b adalah parameter bias.

Pola $\overrightarrow{x_i}$ yang pada kelas +1 (sampel positif) dapat diformulasikan sebagai pola yang dipenuhi oleh pertidaksamaan

$$x_i w + b > 0 \text{ untuk } y_i = 1 \tag{2.4}$$

Pola $\overrightarrow{x_i}$ yang pada kelas -1 (sampel negatif) dapat diformulasikan sebagai pola yang dipenuhi oleh pertidaksamaan

$$x_i w + b < 0 \text{ untuk } y_i = -1 \tag{2.5}$$

Masalah menemukan parameter w dan b yang optimal untuk mendapatkan hiperplane optimal adalah termasuk masalah *quadratic programming*.

$$\min_{w,b} \frac{1}{2} w^T w + C \sum_{i=1}^{m} \mu_i \tag{2.6}$$

Masalah ini dapat diselesaikan dengan berbagi teknik komputasi, termasuk dengan cara menggunakan pengganda *Lagrange*.

$$\sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j \overrightarrow{x_i x_j} \tag{2.7}$$

Di mana α_i adalah pengganda *Lagrange*, dengan nilai nol atau positif ($\alpha_i \geq 0$). Dari hasil perhitungan ini diperoleh α_i yang sebagian besar adalah bernilai positif. Data ini berkorelasi dengan α_i positif tersebut, disebut dengan istilah *Support Vector Machine*.

Pada prinsipnya teknik SVM termasuk klasifikasi linier. Bedanya, teknik SVM memiliki keunggulan dalam klasifikasi pada masalah non-linier. Dalam prosesnya, data diproyeksikan ke ruang vektor baru yang sering disebut ruang fitur (*featur space*), dengan dimensi yang lebih tinggi, sehingga data dapat dipisahkan secara linier. Selanjutnya, di *space* baru, SVM menghitung hyperplane yang optimal, yang berfungsi sebagai pengklasifikasi linier.

Bisa diasumsikan bahwa kedua kelas dapat dipisahkan secara sempurna oleh bidang hiperplane (dapat dipisahkan secara linier/*linear separable*). Akan tetapi, secara umum kedua kelas pada *input space* tidak dapat dipisahkan sepenuhnya (*non*

linear separable). Untuk mengatasinya, SVM diformulasi ulang dengan memperkenalkan teknik *Margin Soft*. Menurut Cortes et al. (1995), konsep margin maksimal yang dimodifikasi memungkinkan untuk contoh *mislabeled*. Jika ada *hyperplane* yang mematahkan contoh "ya" dan "tidak", metode *Soft Margin* akan memilih hyperplane yang membagi *instance* sebersih mungkin. Metode ini memperkenalkan variabel slack ξi ($\xi i \geq 0$), yang mengukur derajat kesalahan klasifikasi.

$$y_i(<\vec{w}.\vec{x_i}>+b) \geq 1-\xi i \quad (i = 1,2,\dots,l) \tag{2.8}$$

Sedangkan fungsi objektif (2.5) yang dioptimasikan menjadi.

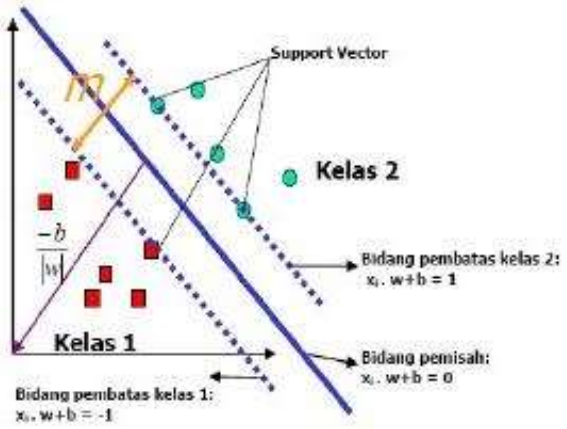
$$\text{minimze}\frac{1}{2}\left|\left|\vec{w}\right|\right|^2 + C\sum_{i=1}^{l}\xi\text{i} \tag{2.9}$$

Parameter C adalah parameter yang mengontrol *trade-off* antara margin dan kesalahan klasifikasi ξ. Semakin besar nilai C maka semakin besar pula skor kesalahan akhir, sehingga proses pelatihan menjadi semakin ketat. Parameter C ditentukan dengan mencoba beberapa nilai dan dievaluasi pengaruhnya terhadap akurasi yang dicapai oleh SVM.

II.4.2. Klasifikasi Data Linear Separable

Data Linear Separable adalah data yang dapat dipisahkan secara linier. Misalkan $\{X1,\dots Xn\}$ adalah kumpulan data (*dataset*) dan $\{+1,-1\}$ adalah label kelas dari data Xi. Pada ***Gambar 2. 1*** menggambarkan berbagai bidang pemisah alternatif

yang dapat memisahkan semua kumpulan data menurut kelasnya. Namun, bidang pemisah terbaik tidak hanya mampu memisahkan data, tetapi memiliki margin terbesar juga (Wu & Kumar, 2009).



Sumber: Wu dan Kumar (2009)

Gambar 2. 1 Hyperlane

II.4.3. Klasifikasi Data Linear Non-Separable

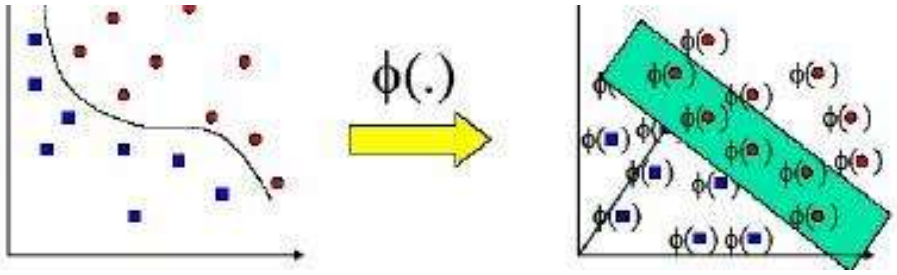
Secara umum, kedua kelas pada ruang masukan (*input space*) tidak dapat dipisahkan dengan sempurna. Untuk mengatasi kondisi tersebut, SVM menggunakan teknik *soft margin*, dengan memasukkan variabel slack $\xi > 0$. Misal ξ yaitu jarak antara *decision boundary* (garis pemisah) dengan data, maka persamaan dirumuskan menjadi:

$$\begin{aligned} \vec{w}.\vec{x_i} + b &\leq -1 + \xi, \text{ jika } y_i = -1 \\ \vec{w}.\vec{x_i} + b &\geq 1 - \xi, \text{ jika } y_i = 1 \end{aligned} \tag{2.10}$$

II.4.4. Klasifikasi Data Non-Linear

Untuk mengatasi masalah *non-linier*, persamaan SVM diubah dengan menyertakan fungsi kernel. Dalam SVM *non-linier*, pertama-tama data $\vec{x}$ dipetakan oleh fungsi $\Phi(\vec{x})$ ke ruang vektor dengan dimensi lebih tinggi. Maka, *hyperplane* yang memisahkan dua kelas dapat dibangun. Selanjutnya, fungsi Φ memetakan setiap data dalam ruang masukan ke ruang vektor baru dengan dimensi yang lebih tinggi (dimensi ke-3), sehingga kedua kelas tersebut dapat dipisahkan secara linier oleh *hyperplane*. Notasi matematis dari pemetaan ini adalah sebagai berikut:

$$\Phi: \mathrm{R}^d \rightarrow \mathrm{R}^d d < q \qquad (2.11)$$



Sumber: Tan *et al* (2006)

Gambar 2. 2 Transformasi dari vektor input ke feature space

Selanjutnya, proses *training* (pembelajaran) pada SVM dalam mencari titik vektor pendukung (*support vector*), tergantung pada perkalian titik (*dot product*) dari data yang telah ditransformasikan ke dalam ruang dimensi baru yang lebih tinggi, yaitu $\Phi(\overrightarrow{x_i}).\Phi(\overrightarrow{x_j})$.

Karena pada umumnya transformasi Φ tidak diketahui, dan sangat sulit dipahami dengan mudah, perhitungan perkalian titik (*dot product*) dapat diganti

oleh fungsi kernel. Secara implisit, $K(\overrightarrow{x_i}, \overrightarrow{x_j})$ mendefinisikan transformasi Φ. Defenisi ini disebut dengan *Kernel Trick*, dirumuskan:

$$K(\overrightarrow{x_i}, \overrightarrow{x_j}) = \Phi(\overrightarrow{x_i}).\Phi(\overrightarrow{x_j}) \quad (2.12a)$$

$$D_{ij} = y_i y_j\left(K(\overrightarrow{x_i}.\overrightarrow{x_j}) + \lambda^2\right) \quad (2.12b)$$

$$E_i = \sum_i^N \alpha_i D_{ij} \quad (2.12c)$$

$$\delta\alpha_i = \min\{\max[\gamma(1 - E_i, -\alpha_i), C - \alpha_i]\} \quad (2.12d)$$

$$\alpha_i = \alpha_i + \delta\alpha_i \quad (2.12e)$$

$$b = -\frac{1}{2}\left(\sum_{i=1}^{n} \alpha_i y_i\left(K(x_i, x^-)\right) + \sum_{i=1}^{n} \alpha_i y_i\left(K(x_i, x^-)\right)\right) \quad (2.12f)$$

$$f\left(\Phi(\vec{x})\right) = \vec{w}.\Phi(\vec{x}) + b$$

$$= \sum_{i=1, x_i \in SV}^{n} \alpha_i y_i K(x, x_i) + b \quad (2.12g)$$

II.4.5. Algoritma SVM untuk Menganalisis Dokumen Web

Langkah-langkah ekstraksi informasi (*Information Extraction* atau IE) adalah proses mengubah dokumen teks yang tidak terstruktur dengan domain tertentu menjadi struktur informasi yang relevan. Domain penelitian ini dalam bahasa Indonesia, dengan menerapkan teknik *machine learning* (pembelajaran mesin). Di sini, pendekatan statistik adalah pendekatan yang digunakan dalam pembelajaran mesin, dengan metode klasifikasi token. Algoritma yang digunakan adalah *Support Vector Machine* (SVM) dengan margin yang tidak rata (*uneven margin*), yang dirancang khusus untuk dataset yang tidak seimbang. (Cawley, 2006).

Parameter dan variabel yang digunakan:

$x = \{x_0, x_1, x_2, \dots, x_m\}$: Sampel Training

$y = \{y_1, \dots, y_m\} \subset \{\pm 1\}$: Label data training

kernel : jenis fungsi kernel

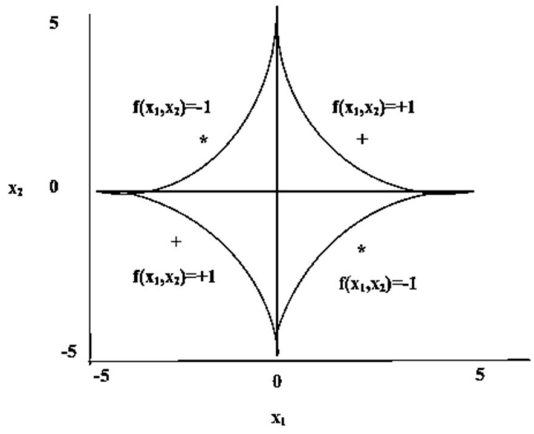
par : parameter kernel

C : konstanta cost

$\alpha = [\alpha_1, \dots, \alpha_m]$: Lagrange multiplier dan bias b

Langkah-langkah yang dilakukan:

1. Cari dan hitung matriks kernel H
2. Menentukan pembatas untuk program kuadratik, termasuk A_{eq}, b_{eq}, A dan b



Sumber: Cawley (2006)

Gambar 2. 3 Ilustrasi data dipisahkan dalam kasus XOR

3. Tentukan fungsi tujuan *quadratic programming* $\frac{1}{2} x\, Hx + f'x$

4. Selesaikan masalah QP

II.4.6. Kelebihan dan Kekurangan SVM Secara Umum

Ada beberapa kelebihan dan kekurangan dari algoritma SVM. Kelebihan dari SVM adalah sebagai berikut (Chowdhury *et al.*, 2018).

1. Efektif jika jumlah dimensi lebih besar daripada jumlah sample
2. Lebih hemat memori karena menggunakan vektor pendukung (support vector)
3. Banyak fungsi kernel yang berbeda yang dapat digunakan untuk klasifikasi

Adapun kekurangan menggunakan algoritma SVM adalah sebagai berikut:

1. Menghindari pemasangan kernel yang berlebihan jika jumlah fitur lebih besar daripada umlah sample
2. Tidak langsung memberikan estimasi probabilitas

II.5. K-Nearest Neighbor

II.5.1. Konsep K-Nearest Neighbor (K-NN)

Algoritma K-Nearest Neighbor (K-NN) merupakan metode pengklasifikasian objek berdasarkan data pembelajaran berdasarkan jarak terdekatnya dengan objek (Sianturi & Sitorus, 2019). Algoritma K-NN merupakan algoritma *supervised learning* di mana hasil *instance query* baru diklasifikasikan berdasarkan mayoritas

kategori pada K-NN, kemudian kelas yang paling banyak muncul adalah kelas hasil klasifikasi. (Banjarsari *et al.*, 2016).

Tujuan algoritma KNN yaitu untuk mengklasifikasikan objek berdasarkan atribut dan sampel pelatihan. Pengklasifikasi tidak menggunakan apa pun untuk dicocokkan dan hanya didasarkan pada memori. Diberikan titik kueri, akan dicari sebanyak *k* objek atau titik pelatihan(*training*) yang paling dekat dengan titik kueri. Klasifikasi menggunakan jumlah suara terbanyak dari antara *k* objek klasifikasi. Algoritma K-Nearest Neighbor (K-NN) menggunakan klasifikasi tetangga (*neighboor*) sebagai nilai prediksi untuk *instance* kueri baru.

Teknik Algoritma K-Nearest Neighbor (K-NN) sangat sederhana, yaitu bekerja berdasarkan jarak terpendek dari *instance query* ke sampel pelatihan untuk menentukan K-NN. Sampel pelatihan diproyeksikan ke dalam ruang multi-dimensi, di mana setiap dimensi mewakili fitur data. Ruang ini dibagi menjadi beberapa bagian berdasarkan klasifikasi sampel pelatihan. Sebuah titik di ruang ini diberi label apabila titik tersebut adalah klasifikasi yang paling banyak yang ditemukan di *k* tetangga terdekat dari titik itu. Teknik menghitung jarak terdapat dua jenis, yaitu metode *Cosine Similarity* atau *Euclidean Distance* yaitu perhitungan jarak terdekat. Perhitungan jarak terdekat diperlukan untuk menentukan jumlah kemiripan yang dihitung dari kemiripan tampilan teks yang dimiliki suatu paragraf. Selanjutnya, tampilan teks yang diujikan dibandingkan dengan masing-masing sampel data asli. Teknik untuk menghitung jarak antar tetangga menggunakan metode *Euclidean Distance* dan *Cosine Similarity* yang direpresentasikan sebagai berikut (Guo *et al.*, 2006):

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{2.13}$$

di mana matriks $D(a, b)$ yaitu jarak skalar antara kedua vektor a dan b dari matriks pada dimensi d.

Berdasarkan nilai N-nya, satuan jenis metode jarak Euclidian ada dua macam yaitu: (Darujati, 2010)

1. 1-NN, yaitu pengklasifikasian dilakukan pada 1 label data terdekat, algoritmanya sebagai berikut:
   - Hitung jarak antara data baru ke setiap label data
   - Tentukan satu pelabelan data yang memiliki jarak minimum
   - Klasifikasi data baru ke dalam pelabelan data
2. k-NN, yaitu pengklasifikasian dengan menentukan nilai pada k label data terdekat, dengan syarat nilai $k > 1$, algoritmanya sebagai berikut:
   - Hitung jarak antara data baru ke setiap label data
   - Tentukan k data pelabelan yang memiliki jarak minimum
   - Klasifikasi data baru menjadi pelabelan data mayoritas.

Metode menghitung jarak *Cosine similarity* yaitu menentukan kecocokan dokumen berdasarkan kueri untuk pengukuran antara vektor dokumen (D) dan vektor kueri (Q). Persamaan *Cosine Similarity* adalah sebagai berikut: (Darujati, 2010)

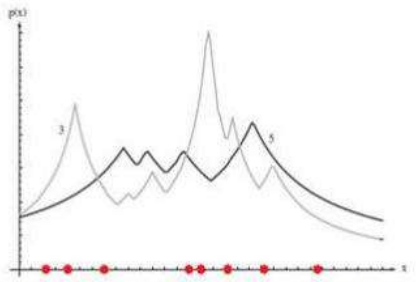
$$\text{sim}(Q, D) = \cos(Q, D) = \frac{Q.D}{|Q|.|D|} \tag{2.14}$$

$$|D| = \sqrt{\left(\sum_{i=1}^{n} D_i^2\right)}$$

$$|Q| = \sqrt{\left(\sum_{i=1}^{n} Q_i^2\right)}$$

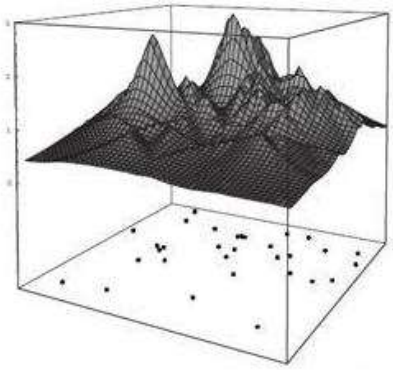
Di mana Q adalah dokumen tes, D adalah dokumen pelatihan, dan Q_i dan D_i adalah bobot nilai yang telah diberikan untuk setiap *term* dalam dokumen.

Pada tahap pelatihan (*training*), algoritma ini hanya menyimpan vektor fitur dan mengklasifikasikan data sampel *training*. Pada tahap klasifikasi, fitur yang sama dihitung untuk data pengujian/*testing* (yang klasifikasinya belum diketahui). Jarak vektor baru ini ke semua vektor sampel *training* dihitung dan jumlah k dari potongan terdekat diambil. Titik yang baru saja diklasifikasikan tersebut diperkirakan termasuk dalam klasifikasi tertinggi dari titik-titik tersebut. Misalnya, untuk memperkirakan $p\ (x)$ dari n sampel *training*, difokuskan pada satu sel di sekitar x dan membiarkannya tumbuh hingga mencakup k sampel. Sampelnya adalah K-NN dari x. Jika kepadatannya tinggi mendekati x, maka sel akan menjadi relatif kecil yang artinya memiliki resolusi yang baik. Jika kepadatannya rendah, sel akan membesar, tetapi akan berhenti setelah memasuki wilayah dengan kepadatan tinggi. Pada ***Gambar 2. 4*** dan ***Gambar 2. 5*** diperlihatkan estimasi kepadatan satu dimensi dan dua dimensi menggunakan K-NN



Sumber: Darujati (2010)

Gambar 2. 4 Delapan titik dalam satu dimensi dan estimasi Kepadatan K-NN dengan $k = 3$ dan $k = 5$



Sumber: Darujati (2010)

Gambar 2. 5 K-NN mengestimasi Kepadatan dua dimensi dengan $k = 5$

Hasil k terbaik untuk algoritma KNN bergantung pada datanya. Secara umum, nilai k yang tinggi akan mengurangi efek *noise* pada klasifikasi, tetapi membuat batas antar setiap klasifikasi semakin kabur. Nilai k yang baik dapat dipilih menggunakan optimasi parameter, contohnya dengan menggunakan validasi silang (*cross validation*). Kasus khusus di mana melakukan prediksi klasifikasi berdasarkan data pelatihan terdekat (dengan kata lain, $k = 1$) disebut algoritma

ketetanggaan terdekat (*nearest neighbor*). Tingkat akurasi algoritma K-NN sangat dipengaruhi oleh ada atau tidaknya fitur yang tidak relevan atau apabila bobot fitur tidak setara dengan relevansinya dengan klasifikasi.

II.5.2. Algoritma K-NN untuk Menganalisis Dokumen Web

Berikut merupakan teknik untuk menghitung *K-Nearest Neighbors* pada kasus dokumen teks (Chuang & Wu, 2004):

1. Menentukan parameter k sebagai jumlah tetangga terdekat yaitu menggunakan $k = 1$, sehingga apabila ditemukan tetangga terdekat, defenisikan sebagai nilai prediksi.
2. Menghitung jarak antara data yang masuk dan semua sampel pelatihan yang ada. Dalam penelitian ini jenis jarak terdekat yang digunakan adalah *cosine similarity* pada persamaan (2.14).
3. Urutkan objek ke dalam kelompok yang memiliki jarak terkecil.
4. Kumpulkan kategori Y (klasifikasi tetangga terdekat).
5. Berdasarkan kategori mayoritas, nilai hitung dari *query instance* dapat diprediksi, kemudian ditentukan jarak ke tetangga terdekat yang akan dijadikan nilai prediksi dari data selanjutnya.

II.5.3. Kelebihan dan Kekurangan Algoritma K-NN Secara Umum

Setiap algoritma pasti memiliki kelebihan dan kekurangan dalam setiap cara kerjanya begitu juga dengan algoritma ini. Adapun kelebihan dari algoritma K-NN adalah sebagai berikut:

1. Efektif digunakan pada data training yang besar
2. Data yang dihasilkan lebih akurat
3. Cocok digunakan pada data training yang mengandung noise
4. Dapat digunakan pada klasifikasi non-linear

Sedangkan kekurangan dari algoritma ini yaitu perlu ditentukan dahulu nilai k yang paling optimal sebelum melakukan klasifikasi.

II.6. Text Preprocessing

Menurut Feldman *et al*. (2007), *text preprocessing* (preprocessing teks) adalah langkah awal dari teks untuk menyusun teks menjadi data yang akan diolah lebih lanjut (Feldman & Sanger, 2007). Teks yang ada harus dipisahkan, ini dapat dilakukan pada beberapa tingkatan yang berbeda. Sebuah dokumen dapat dipecah menjadi beberapa bab, sub-bab, paragraf, kalimat yang kemudian menjadi potongan kata/token. Selain itu, pada tahap ini, keberadaan angka numerik, huruf kapital, atau karakter lain dihilangkan dan diubah. Berikut proses tahapan preprocessing teks:

1. Tokenisasi/*Tokenizing*

   Tahap tokenisasi yaitu pemotongan string input berdasarkan setiap kata yang menyusunnya.

2. *Case Folding*

   Sebagaimana dijelaskan sebelumnya, bahwa pada tahap preprocessing teks beberapa hal akan diubah, semua huruf pada dokumen diubah menjadi huruf kecil. Huruf yang dimaksud adalah dari "a" sampai "z". Karakter diluar huruf, dihilangkan dan dianggap sebagai pembatas/*delimiter*.

3. *Cleaning*

   Tahap ini adalah proses membersihkan dokumen dari beberapa kata yang tidak perlu atau *unnecessary* sehingga *noise* pada teks berkurang pada tahap klasifikasi. Biasanya, kata ataupun karakter yang dihilangkan pada data komentar diantaranya simbol-simbol, numerik, alamat URL, tanda pagar (#), dan mention (@username).

4. Normalisasi

   Tahap ini adalah proses untuk mengubah kata-kata non-standar. Proses penanganan kata seperti ini dilakukan didasarkan pada kamus yang terdiri dari kata nonstandar beserta kata standarnya. Pada Bahasa Indonesia misalnya seperti kata-kata “alay”.

5. *Filtering*

   Tahap ini adalah menghilangkan kata umum yang biasanya muncul dalam jumlah besar dan dianggap tidak berarti, karena memberikan pengaruh yang kecil. Tahap ini biasa disebut penghapusan *stopword*. Beberapa contoh kata yang dikategorikan ke dalam *stopwords* yang biasanya muncul dalam jumlah besar dan tidak memiliki arti, antara lain "di", "dan", "dengan", "ke", indikator waktu, dan kata-kata tanya.

6. *Stemming*

   Tahapan ini adalah tahapan mencari kata dasar dari setiap kata yang di-*filter*. Pada tahap ini, proses pengembalian berbagai bentukan kata dilakukan ke dalam representasi yang sama. Dalam Bahasa Indoensia, biasanya menggunakan kamus Sastrawi.

II.7. Pembobotan TF-IDF

TF (*Term Frequency*) merupakan proses pembobotan suatu kata yang dianggap proporsional dengan banyaknya kemunculan kata dalam suatu dokumen, dari TF diperoleh DF (*Document Frequency*) yaitu pembobotan untuk mengukur penting tidaknya suatu kata dalam dokumen tersebut. (Hardiyanto *et al.*, 2016). Pembobotan TF-IDF dihitung dengan hasil perkalian antara TF dan IDF, di mana IDF adalah hasil invers dari DF. Adapun persamaan TF-IDF sebagai berikut:

$$IDF(w) = \log\left(\frac{N}{DF(w)}\right) \tag{2.15}$$

$$W\ dt = TF \cdot IDF \tag{2.26}$$

$$TF - IDF(w,d) = \frac{TF - IDF(w,d)}{\sqrt{\sum_{w=1} n\, TF - IDF(w,d)^2}} \tag{2.17}$$

Keterangan:

$IDF(w)$	:	bobot kata pada semua dokumen
w	:	Satu kata
$TF(w,d)$	:	Frekuensi kemunculan kata w pada dokumen d

$IDF(w, d)$: invers DF dari kata w

N : jumlah semua dokumen

$DF(w)$: jumlah dokumen yang memuat kata w

II.8. Pengujian

Dalam pengujian algoritma, *Confussion Matrix* dilakukan untuk pengujian akurasi algoritma. Di bidang data mining, untuk melakukan perhitungan akurasi suatu metode digunakan *confussion matrix*. Pada dasarnya *Confussion Matrix* berisi informasi yang membandingkan antara hasil klasifikasi sistem dengan hasil klasifikasi sebenarnya (*actual*) (Utomo *et al.*, 2014). Dalam *confusion matrix*, ada jenis klasifikasi biner yang hanya memiliki 2 output kelas sebagai berikut:

Tabel 2. 1 Klasifikasi *Binary*

Kelas/*Class*	**Terklasifikasi Positif**	**Terklasifikasi Negatif**
Sebenarnya Positif	TP (*True Positive*)	FP (*False Positive*)
Sebenarnya Negatif	FN (*False Negative*)	TN (*True Negative*)

Perhitungan *Matrix Confusion* menghasilkan 4 keluaran, yaitu *recall*, presisi (*precision*), akurasi (*accuracy*), dan tingkat kesalahan (*error rate*). *Recall* adalah tingkat keberhasilan sistem dalam memulihkan informasi. *Presisi* merupakan tingkat kebenaran antara informasi yang diharapkan oleh pengguna dan jawaban

oleh sistem. Sedangkan akurasi diartikan sebagai tingkat kedekatan antara nilai prediksi dengan nilai sebenarnya. Persamaan *Confusion Matrix* yaitu:

$$recall = \frac{TP}{FN + TP} \times 100\% \quad (2.18)$$

$$precission = \frac{TP}{FP + TP} \times 100\% \quad (2.19)$$

$$accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \times 100\% \quad (2.20)$$

Keterangan:

TP (True Positive) : banyaknya data positif yang terklasifikasi benar

TN (True Negative) : banyaknya data negatif yang terklasifikasi benar

FN (False Negative) : banyaknya data negatif yang terklasifikasi salah

FP (False Positive) : banyaknya data positif yang terklasifikasi salah

II.9. Penelitian Relevan

Adapun beberapa penelitian terdahulu yang relevan dengan penelitian ini dapat dilihat pada tabel berikut:

Tabel 2. 2 Daftar Penelitian Relevan

No	Peneliti (Tahun)	Judul Penelitian	Metode Analisis	Isi Penelitian
1	Trinh, Son; Nguyen, Luu; Vo,	Lexicon Based Sentiment Analysis of	Lexicon Based, Support Vector Machine (SVM)	Kesamaan pada penelitian ini adalah Menggunakan

	Minh; Do, Phuc (2016)	Facebook Comments in Vietnamese Language		Support Vector Machine (SVM) untuk mengklasifikasikan kalimat.
2	Fang, Xing; Zhan, Justin (2015)	Sentiment analysis using product review data	Kategorisasi polaritas sentimen Menentukan kalimat positif, negatif dan netral dengan level-document, level-sentence dan entiti and aspect level. Menggunakan POS tagging untuk mengetahui besar jumlah sentimen pada kata. Menggunakan metode Support Vector Machine, Random Forrest dan Naïve Bayesian dalam label kalimat Untuk membandingkan	• Penelitian ini membahas tentang analisis sentimen pada beberapa ulasan produk yang data nya diambil dari situs online Amazon.com. Ulasan yang diambil dari ulasan bintang 1 - 5.

			metode mana yang terbaik dalam memprediksi kalimat. Menggunakan sckit learn	
3	Eung Woo Cho, Moon Soo Cha, So Yeon Kim, Joo Cheol Song, Kyung-Ah Sohn (2014)	Investigating Temporal and Spatial Trends of Brand Images using Twitter Opinion Mining	• Menggunakan Algoritma klasifikasi Naïve Bayes dan Support Vector Machine	• Menambah nilai dari sentiment terhadap peningkatan akurasi klasifikasi dengan melakukan seleksi dan eliminasi *term* yang akan dimodelkan • Membuat kamus data untuk bahasa korea
4	Yu Zhang & Pedro Desouza (2014)	Enhance The power of Sentiment Analysis	Menggunakan teknik transformasi data set ke bentuk yang seragam untuk mempercepat proses data mining dan melakukan perbandingan 5 algoritma klasifikasi	• Menghasilkan metode yang dapat digunakan untuk beberapa bentuk review dari twitter, customer review, facebook dan google+
5	Dhaoui, Chedia;	Social media sentiment	membandingkan 2 metode yaitu	Dataset yang digunakan adalah beberapa komentar

	Webster, Cynthia M.; Tan, Lay Peng (2017)	analysis: lexicon versus Machine learning	lexicon-based dengan machine learnin	facebook terhadap suatu produk. Menggunakan pendekatan machine learning dalam pengujian data berupa fitur ngram