BAB II

LANDASAN TEORI

2.1 *Climate Change*

Perubahan iklim atau *climate change* adalah berubahnya kondisi fisik atmosfer bumi antara lain suhu dan distribusi curah hujan yang membawa dampak luas terhadap berbagai sektor kehidupan manusia (Kementerian Lingkungan Hidup, 2001). Perubahan iklim merupakan sesuatu yang sulit untuk dihindari dan dapat memberikan dampak pada berbagai segi kehidupan. Dampak ekstrem yang terjadi adalah terjadinya kenaikan temperatur dan juga bergesernya musim.

Jutaan tahun yang lalu, sebagian wilayah dunia yang kini lebih hangat dahulunya merupakan wilayah yang tertutupi es, dan beberapa abad terakhir ini, suhu rata-rata telah naik turun secara musiman. Pemanasan ekstrem semakin intens sejak 1950-an. Jika diamati berdasarkan gambar 1.1 yaitu grafik perubahan suhu global (IPCC, 2021), tingkat pemanasan global meningkat dua kali lipat dalam 50 tahun terakhir. Temperatur rata-rata global naik sebesar 0.74^0C selama abad ke-20, sedangkan pada lima tahun terakhir merupakan suhu terpanas dalam sejarah sejak 1850. Hal ini semakin menjelaskan bahwa pemasan global telah menjadikan perubahan iklim yang merubah banyak sistem penunjang bumi.

Perubahan iklim salah satunya disebabkan akibat fruktuasi radiasi matahari atau akibat legusan gunung berapi secara berkala. Namun yang baru adalah bahwa perubahan iklim bukan hanya oleh peristiwa alam melainkan lebih karena aktivitas manusia. Dalam publikasi IPCC (2021) keadaan bumi saat ini tidak dipungkiri

bahwa manusia terlibat dalam meningkatnya suhu di atmosfer, lautan dan daratan. Kemajuan pesar pembangunan ekonomi Indonesia memberikan dampak yang serius terhadap iklim dunia, antara lain lewat pembakaran secara besar-besaran batu bara, minyak serta pembabatan hutan. Kerusakannya terutama terjadi melalui produksi gas rumah kaca. Sebuah laporan yang disponsori oleh Bank Dunia dan pemerintah Inggris menyatakan Indonesia sebagai negara penghasil emisi gas rumah kaca terbesar ketiga di dunia (Mulyani, 2009). Total emisis gas Indonesia tersebut 85% berasal dari sektor kehutanan yang terutama karena proses deforestasi.

Kehancuran hutan Indonesia berlangsung makin cepat saja, yaitu dari 600.000 hektar per tahun pada tahun 1980an menjadi sekitar 1,6 juta hektar per tahun di penghujung tahun 1990an. Akibatnya, tutupan hutan menurun secara tajam, dari 129 juta hektar pada tahun 1990 menjadi 82 juta ditahun 2000, dan diproyeksikan menjadi 68juta hektar ditahun 2008, sehingga kini setiap tahun Indonesia semakin mengalami penurunan daya serap karbon dioksida (UNDP, 2007).

Dengan meningkatnya emisis dan berkurangnya penyerapan, tingkat gas rumah kaca di atmosfer kini menjadi lebih tinggi ketimbang yang pernah terjadi didalam catatan sejarah. Badan dunia yang bertugas memnitori isu ini *Intergovermental Panel on Climate Change* (IPCC) telah memperkirakan bahwa antara tahun 1950 dan 2005 konsentrasi karbon dioksida di atmosfer meningkat dari sekitar 280 ppm (*part per million)* menjadi 379 ppm per tahun. Sejak saat itu terjadi peningkatan dengan kecepatan 1,9 ppm per tahun (Perdana, 2015). Gas rumah kaca

tersebut memungkinkan sinar matahari menembus atmosfer bumi sehingga menghangatkan bumi, tetapi gas tersebut mencegah pemantulan kemali sebagian udara panas ke ruang angkasa. Akibatnya bumi dan atmosfer perlahan-lahan memanas.

Menurut Murdiyarso (2007 dalam LAPAN) kajian IPCC 4AR yang menyatakan iklim di Indonesia secara spesifik, antara lain : meninkatkan hujan dikawasan utara dan menurunnya hujan di selatan (khatulistiwa), kebakaran hutan dan lahan yang berpeluang besar dengan meningkatnya frekuensi dan intensitas El-Nino (gejala penyimpangan anomali pada suhu permukaan Samudra Pasifik di pantai Barat Ekuador dan Peru yang lebih tinggi daripada rata-rata normalnya), Delta sungai Mahakam masuk ke dalam peta kawasan pantai yang rentang.

2.2 *Twitter*

Twitter diciptakan oleh Jack Dorsey di tahun 2006 dan pertama diluncurkan di dunia maya saat juli 2006 dengan alamat http://www.twitter.com yang masih digunakan hingga saat ini. Pengguna *twitter* dapat menulis pesan berdasarkan topik dengan menggunakan tanda # (*hashtag).* Sedangkan untuk menyebutkan atau membahas pesan dari pengguna lain bisa menggunakan tanda @.

Kegunaan *twitter* selain sebagai media untuk berbagi informasi dengan mem-*post* berbagai macam *tweet, twitter* juga kerap sering digunakna untuk bersosialisasi antar pengguna dan mengungkapkan sentimen atau opini mereka terhadap suatu topik atau isu-isu yang sedang hangat diperbincangkan, tidak hanya opini yang positif tetapi juga opini negatif. Kebiasaan tersebut terkadang tidak dianggap begitu penting untuk disikapi karena telalu banyak *tweet.* Namun kebiasaan tersebut begitu

penting apabila dipelajari lebih lanjut karena kabiasaan tersebut bisa dimanfaatkan juga sebagai sebuah analisa sentimen atau opini terhadap isu-isu yang hangat dan dapat dijadikan informasi untuk mengetahui sentimen atau opini masyarakat Indonesia pada permasalahan tertentu (Aji dan Hidayatullah, 2019).

Ada beberapa cara untuk mendapatkan sekumpulan data *tweet* pada *twitter* yang salah satunya dengan menggunakan *twitter* API. *Twitter* memiliki *Aplication Programming Interface* (API) sedemikian hingga *developer* dapat mengembangkan aplikasi sesuai dengan kebutuhannya masing-masing (Aji, 2019). Berdasarkan pada situs resmi *twitter*, sebuah API merupakan salah satu cara *computer* "berbicara" satu sama lain sehingga dapat memesan dan mengantar informasi. Pengguna dapat dapat mengakses bermacam-macam *tweet* dengan menggunakan suatu kata kunci tertentu. Menurut Monarizqa *et al* (2014) API biasa digunakan untuk penggalian data karena melalui API ini dinformasi bisa didapatkan secara *realtime* dengan *volume* yang sangat tinggi. Dokumentasi mengenai *twitter* API dapat dilihat pada http://dev.twitter.com. Ada beberepa jenis *twiter* API seperti:

1. *Twitter Rest* API

   Terdiri dari *Twitter Rest* dan *Twitter Search. Twitter Rest* memberikan *core* data dan *core twitter objects. Twitter Search* berfungsi untuk melakukan pencarian mengenai suatu *instance* objek *twitter* maupun mencari *trend.*

2. *Twitter Streaming* API

   API ini biasanya digunakna untuk penggalian data karena melalui API ini informasi bisa didapatkan secara *realtime* dengan volume yang sangat tinggi.

2.3 Analisis Sentimen

Analisis sentimen merupakan bidang studi untuk menganalisa pendapat, pandangan, evaluasi, penilaian, sikap, dan emosi terhadap aspek-aspeknya yang diekspresikan melalui teks. Tujuan utama dari analisis sentimen yaitu untuk mengukur perspektif , sentimen, evaluasi, sikap dan emosi pembicara atau penulis berdasarkan perlakuan komputasi subjektivitas dalam sebuah teks (Amalina, 2020).

Analisis sentimen sebagai tugas klasifikasi karena mengklasifikasikan orientasi teks menjadi positif atau negatif. Hal-hal penting yang berhubungan dengan analisis sentimen berdasarkan pengertiannya antara lain :

1. Opini dan Sentimen

   Istilah pendapat dalam analisis sentimen (*opinion mining)* dapat diwakili dengan *quadruple* (s, g, h, t) yang meliputi empat komponen yaitu : orientasi sentimen s, target sentimen g, pemegang pendapat h, dan waktu t. Sentimen mewakili perasaan, sikap, evaluasi atau emosi terkait dengan pendapat. Orientasi sentimen meliputi positif, negatif atau netral. Target sentimen adalah entitas atau aspek entitas yang telah diungkapkan berdasarkan sentimen. Pemegang pendapat adalah individu atau organisasi yang memegang pendapat. Waktu adalah ketika pendapat dinyatakan.

2. Klasifikasi

   Teknik klasifikasi merupakan teknik untuk memprediksi variabel target berdasarkan variabel *input.* Prediksi dari klasifikasi didasarkan pada model yang membangun dari kumpulan data sebelumnya yang telah diketahui. Intinya klasifikasi dalam analisis sentiment adalah proses untuk menentukan kelas atau

nilai dari suatu objek (dokumen) berdasarkan kelas yang telah ditentukan sebelumnya. Prasetyo (2012) mengungkapkan adanya dua pekerjaan utama dalam klasifikasi, yaitu :

1. Pembangunan model sebagai prototipe untuk disimpan sebagai memori.
2. Penggunaan model tersbut untuk melakukan pengenalan/ klasifikasi/ prediksi pada suatu objek data lain agar diketahui di kelas mana objek data tersebut dalam model yang sudah disimpan.

2.4 *Text Mining*

Text mining adalah salah satu bidang khusus dalam *data mining* yang memiliki definisi menambang data berupa teks dimana sumber data biasanya didapatkan dari dokumen dengan tujuan mencari kata-kata yang dapat mewakili isi dari dokumen sehingga dapat dilakukan analisa keterhubungan antar dokumen. Feldman *et al* (2007) menerangkan bahwa *text mining* merupakan proses penambangan yang dilakukan oleh komputer untuk mendapatkan sesuatu yang baru, sesuatu yang tidak diketahui sebelumnya untuk menemukan kembali informasi yang tersirat secara implisit, yang berasal dari informasi secara otomatis dari sumber-sumber data teks yang berbeda.

Sumber data yang digunakan pada *text mining* adalah sekumpulan teks yang memiliki format yang tidak terstruktur atau minimal semi terstuktur. Hal ini menyebabkan adanya tantangan tambahan pada *text mining* yaitu struktur teks yang kompleks dan tidak lengkap, arti yang tidak jelas dan tidak standar, dan bahasa yang berbeda ditambahi translasi yang tidak akurat (Sasmita dan Falani, 2018).

Text mining bertujuan untuk memperoleh informasi yang berguna dari sekumpulan dokumen yang diklasifikasikan secara otomatis. Selain itu, *text mining* mempunyai tujuan untuk mencari kata dalam sekumpulan dokumen dan melakukan analisa keterhubungan kata dalam dokumen tersebut (Praseptian dan Indriani, 2014). Konsepnya, *text mining* biasanya digunakan dalam klasifikasi dokumen tekstual dimana dokumen-dokumen tersebut akan diklasifikasikan sesuai dengan topik dokumen tersebut. Kata-kata yang dapat mewakili isi dari dokumen tersebut dianalisa dan dicocokkan pada basis data kata kunci yang telah ditentukan sebelumnya. Sehingga dengan adanya *text mining* dapat membantu melakukan pengelompokkan suatu dokumen dengan waktu yang singkat.

Langkah-langkah yang dapat dilakukan dalam melakukan *text mining* adalah sebagai berikut :

2.4.1 *Text Preprocessing*

Teknik *preprosessing* diterapkan pada kumpulan data target untuk mengurangi ukuran dan untuk meningkatkan efektivitas sistem *information retrieval* (Amalina, 2020). Dalam bidang *text mining*, *text preprocessing* digunakan untuk mengekstraksi data yang tidak terstruktur. Pengambilan *information retrieval* sebagian besar merupakan masalah untuk menentukan informasi atau dokumen mana yang harus diambil, dimana informasi tersebut sesuai dengan apa yang dibutuhkan pengguna. Informasi yang dibutuhkan pengguna diwakili oleh *query* yang berisi satu atau lebih istilah pencarian, juga dengan beberapa informasi tambahan seperti bobot setiap kata. Keputusan dari data mana yang harus diambil dibuat dengan membandingkan istilah *query* dengan kata atau frasa yang penting

yang muncul dalam dokumen. Sebelum data terambil dari dokumen, terlebih dahulu harus dilakukan pengolahan data karena adanya varian stuktural didalam kata yang nampak pada dokumen atau *query.*

Berkut adalah tahap proses secara umum yang dilakukan dalam tahapan awal untuk *text processing* adalah sebagai berikut :

1. *Cleaning*

   Pada tahap ini *cleaning* dilakukan untuk membersihkan dokumen dengan cara menghilangkan data yang tidak digunakan. Aditya (2015) mencontohkan data yang perlu dihilangkan seprti menghapus karakter HTML, ikon emosi, *username* (@username*)*, *hashtag (#),* url (http://situs.com), dan email (nama@situs.com).

Tabel 1.1 Contoh Luaran *Cleaning*

Tweet	Luaran
b'RT @ICRC_id: Kita semua punya peran untuk memerangi perubahan iklim. \n\n#ClimateActionNow \n\nhttps://t.co/uNDskdfXtC'	Kita semua punya peran untuk memerangi perubahan iklim
b'Sepertinya IKN merupakan program untuk mendukung percepatan perubahan iklim bumi'	Sepertinya IKN merupakan program untuk mendukung percepatan perubahan iklim bumi

2. *Drop Duplicates*

Tweet yang masuk dalam *database* terdapat *tweet* yang mempunyai teks yang sama, oleh karena itu diperlukan *preprosessing duplicates* dengan tujuan mereduksi *tweet.*

Tabel 1.2 Contoh Luaran *Drop Duplicates*

Data Frame	Luaran
4.473	1.052

3. *Case Folding*

Case Folding merupakan proses dari penyamaan *case* dalam sebuah dokumen. Tidak semua dokumen mentah yang berbentuk teks konsisten dalan penggunaan huruf kapital, oleh karena itu perlu dilakukan proses untuk pengubahan bentuk data teks tersebut menjadi konsisten. Pada tahap ini *case folding* dibutuhkan untuk mengonversi seluruh teks dalam sebuah dokumen menjadi benruk standar (dalam hal huruf kecil atau *lowercase).*

Tabel 1.3 Contoh Luaran *Case Folding*

Tweet	Luaran
Sepertinya IKN merupakan program untuk mendukung percepatan perubahan iklim bumi	sepertinya ikn merupakan program untuk mendukung percepatan perubahan iklim bumi
Kita semua punya peran untuk memerangi perubahan iklim	kita semua punya peran untuk memerangi perubahan iklim

4. *Labelling*

Pemberian label suatu pernyataan yang diklasifikasikan menjadi sentimen positif dan negatif. *Labelling* ke dalam kategori positif dan negatif ditentukan menggunakan kumpulan kata dengan bahasa Indonesia yang terdiri dari kumpulan kata-kata positif dan negatif.

Tabel 1.4 Contoh Luaran *Labelling*

Tweet	Labelling
kita semua punya peran untuk memerangi perubahan iklim	Positif
sepertinya ikn merupakan program untuk mendukung percepatan perubahan iklim bumi	Negatif

5. *Tokenizing*

Tokenizing adalah proses memecah kalimat menjadi kata-kata yang dilakukan untuk menjadikan sebuah kalimat menjadi bermakna. *Tokenizing* merupakan istilah yang digunakan dalam proses pemecahan atau pemisahan kata.

Tabel 1.5 Contoh Luaran *Tokenizing*

Tweet	*Tokenizing*
kita semua punya peran untuk memerangi perubahan iklim	['kita', 'semua', 'punya', 'peran', 'untuk', 'memerangi', 'perubahan', 'iklim']

sepertinya ikn merupakan program untuk mendukung percepatan perubahan iklim bumi	['sepertinya', 'ikn', 'merupakan', 'program', 'untuk', 'mendukung', 'percepatan', 'perubahan', 'iklim', 'bumi']

6. *Spelling Normalization*

Spelling Normalization merupakan proses atau subtitusi kata-kata yang salah eja atau singkatan dalam bentuk tertentu. Subtitusi sebuah kata yang dilakukan untuk menghindari jumlah perhitungan dimensi kata yang melebar. Inti dari tahap ini yaitu penggantian kata yang sesuai dengan KBBI dari penulisan kata yang berlebihan atau disingkat.

Tabel 1.6 Contoh Luaran *Spelling Normalization*

Tweet	*Luaran*
hindari global warming dgn menebang hutan ilmu dari mana itu	['hindari', 'global', 'warming', 'dengan', 'menebang', 'hutan', 'ilmu', 'dari', 'mana', 'itu']
takut uga ini climate change minggu kmrn panasnya mana ada trs skrng tiba dingin	['takut', 'juga', 'ini', 'climate', 'change', 'minggu', 'kemarin', 'panasnya', 'mana', 'ada', 'terus', 'sekarang', 'tiba', 'dingin']

7. *Stemming*

Stemming merupakan istilah untuk proses pencarian *root* atau akar dari kata yang dihasilkan oleh proses *filtering*. Hasil indeks dapat diperkecil dengan mencari kata dasar atau *root* tanpa harus kehilangan arti kata tersebut. Tahap *stemming* pada *text preprocessing* dilakukan untuk mengubah kata-kata yang berimbuhan menjadi kata dasar sesuai dengan KBBI. Imbuhan kata yang perlu dihilangkan yaitu berupa awalan, akhiran dan kombinasi.

Tabel 1.7 Contoh Luaran *Stemming*

Tweet	Luaran
['takut', 'juga', 'ini', 'climate', 'change', 'minggu', 'kemarin', 'panasnya', 'mana', 'ada', 'terus', 'sekarang', 'tiba', 'dingin']	['takut', 'juga', 'ini', 'climate', 'change', 'minggu', 'kemarin', 'panas', 'mana', 'ada', 'terus', 'sekarang', 'tiba', 'dingin']
['hindari', 'global', 'warming', 'dengan', 'menebang', 'hutan', 'ilmu', 'dari', 'mana', 'itu']	['hindar', 'global', 'warming', 'dengan', 'tebang', 'hutan', 'ilmu', 'dari', 'mana', 'itu']

8. *Fitlering Stopwords*

Stopwords adalah kata yang sering muncul namun tidak mengandung kontribubsi apapun terhadap makna dokumen (Galioutou *et al*, 2013). Proses *filtering stopwords* merupakan proses dalam memilih kata-kata yang nantinya digunakna untuk mewakili dokumen. Oleh karena itu kata-kata yang tidak termasuk ke dalam *stopwords* perlu dihilangkan agar proses klasifikasi lebih cepat.

Tabel 1.8 Contoh Luaran *Fitlering Stopwords*

Tweet	*Stopwords*
['takut', 'juga', 'ini', 'climate', 'change', 'minggu', 'kemarin', 'panas', 'mana', 'ada', 'terus', 'sekarang', 'tiba', 'dingin']	['takut', 'climate', 'change', 'minggu', 'kemarin', 'panas', 'dingin']
['hindar', 'global', 'warming', 'dengan', 'tebang', 'hutan', 'ilmu', 'dari', 'mana', 'itu']	['hindar', 'global', 'warming', 'tebang', 'hutan', 'ilmu']

2.4.2 *Wordcloud*

Wordcloud merupakan sebuah sistem yang memunculkan visualisasi kata-kata dengan memberikan penekanan pada frekuensi kemunculan kata terkait dalam wacana tertulis (Qeis, 2017). Secara umum *wordcloud* adalah representasi visual dari data teks, biasa digunakan untuk menggambarakan data pada sebuah situs.

Hasil klasifikasi analisis sentimen data teks juga dapat direpresentasikan secara visual melalui *wordcloud*, dimana *wordcloud* merupakan tampilan dari kata-kata yang sering muncul dari data teks hasil *crawling*. Tampilan *wordcloud* dapat dibedakan sesuai kategori label sentimen yang digunakan pada penelitian klasifikasi analisis sentimen.

Gambar 1.1 Contoh *Wordcloud*

2.5 *TF-IDF*

Karena setiap kata memiliki tingkatan kepentingan yang berbeda dalam dokumen, maka untuk setiap kata tersebut diberikan sebuah indikator, yaitu *term weight*. *Term wight* atau pembobotan *term* Zafikri (2008) dipengaruhi oleh hal berikut :

1. *Term Frequency* (TF)

   TF adalah faktor yang menentukan bobot *term* pada suatu dokumen berdasarkan jumlah kemunculan dalam dokumen tersebut. Nilai jumlah kemunculan suatu *term*/kata diperihtunkan dalam pemberian bobot terhadap suatu kata. Semakin besar jumlah kemunculan suatu *term* (tf_{ad} tinggi) dalam dokumen, maka bobot dalam dokumen akan memberikan nilai yang semakin besar.

2. *Inverse Dokument Frequency* (IDF)

   IDF adalah pengurangan dominasi *term* yang sering muncul di berbagai dokumen. Hal ini diperhatikan karena *term-term* yang banyak muncul diberbagai dokumen, dapat dianggap sebagai *term* umum sehingga tidak

penting nilainya. IDF menghitung nilai log dari pembagian antara total jumlah dokumen dalam suatu *corpus* dengan jumlah dokumen yang mengandung *term* tertentu.

$$idf_a = \log \frac{D}{Df_a} \quad (1.1)$$

Keterangan :

D = Jumlah dokumen dalam suatu *corpus*

Df_a = Jumlah dokumen yang mengandung kata *a*

TF-IDF (*Term Frequency and Inverse Document Frequency*) adalah suatu proses dari teknik ekstraksi fitur dengan proses memberikan nilai pada masing-masing kata yang ada pada *data training*. Untuk mengetahui seberapa penting sebuah kata mewakili sebuah kalimat, akan diberi perhitungan. Fitur TF-IDF digunakan untuk menghitung nilai (*W*) dari masing-masing dokumen terhadap kata kunci dengan formula pada persamaan berikut:

$$W_{ad} = TF_{ad} \; x \; IDF_a \quad (1.2)$$

Keterangan :

W_{ad} = Nilai atau bobot dokumen *d* pada kata *a*

TF_{ad} = Jumlah kata *a* yang dicari dalam suatu dokumen *d*

IDF_a = *Inverse dokument frequency* dari kata *a*

2.6 *Naïve Bayes Classifier*

Naïve Bayes Classifier merupakan pengklasifikasian dengan probabilitas dan statistik yang dikemukakan oleh ilmuwan Inggris Thomas Bayes yaitu memprediksi

peluan dimasa depan berdasarkan pengalaman dimasa sebelumnya (Bustami, 2014). Secara umum teorema Bayes dapat dinotasikan pada persamaan berikut :

$$P(v|a) = \frac{P(v)\,P(a|v)}{P(a)} \qquad (1.3)$$

Keterangan :

v = target label (kelas)

a = kata dengan kelas yang belum diketahui

$P(v|a)$ pada algoritma diatas (2.3) merupakan pribabilitas dari kejadian *v* jika diketahui kejadian *a* muncul. $P(v)$ adalah *prior*. *Prior* merupakan nilai probabilitas dari kemunculan suatu nilai target label tertentu tanpa memperhatikan nilai *features* (variabel prediktor). $P(a|v)$ adalah *likelihood*. *Likelihood* yaitu probabilitas kemunculan nilai *feature* tertentu apabila diketahui kemunculan nilai terget labelnya. $P(a)$ disebut sebagai *evidence*, yaitu probabilitas dari kemuculan sekumpulan nilai *features* (variabel prediktor).

Algoritma *naïve bayes classifier* merupakan algoritma yang sering digunakan dalam klasifikasi analisis sentimen pada jumlah data yang besar dan menghasilkan akurasi yang tinggi serta tahap *processing* klasifikasi juga bekerja dengan baik dan cepat (Ratnawati, 2018).

Algoritma *Naïve Bayes Classifier* bertujuan untuk mencari klasifikasi dari data yang akan diujikan dengan mencari nilai propabilitas tertinggi dalam pengujian (Permana dan Effendi, 2019). Dalam algoritma *naïve bayes classifier* setiap dokumen direpresentasikan dengan pasangan atribut "a_1, a_2,…, a_n" dimana a_1 adalah kata pertama, a_2 adalah kata kedua dan seterusnya. Sedangkan *v* adalah himpunan kategori *tweet*. Pada tahap klasifikasi algoritma akan mencari probabilitas tertinggi

dari semua kategori-kategori yang diujikan dengan *naïve bayes classifier* dengan simbol V_{MAP} dan dituliskan sesuai dengan persamaan berikut:

$$V_{MAP} = armax_{vj \epsilon v} \prod_{a=1}^{n} (P(a_i|v_j)\, P(v_j)) \qquad (1.4)$$

Keterangan :

a = Kata yang belum terklasifikasi

v = Kategori kelas dalam data *training*

$P(v_j)$ = Probabilitas kelas j

$P(a_i|v_j)$ = Probabilitas kata a_i kelas v_j

Metode *naïve bayes* mempunyai 2 (dua) tahapan dalam proses klasifikasi teks, yaitu proses pelatihan (*training)* dan proses klasifikasi. Klasifikasi yang dimaksud yaitu mendapatkan kesimpulan berupa opini positif atau negatif dari analisis sentimen yang diuji. Dalam buku *An Intoduction to Information Retrieval* yang dituliskan oleh Manning *et al* (2009) tahapan pelatihan (*training*) sebagai berikut:

1. Menghitung *prior* setiap kategori $P(v_j)$

$$P(v_j) = \frac{n_j}{n_{doc}} \qquad (1.5)$$

Dimana :

n_j = Jumlah *tweet* dalam *data training* dengan kelas *j*

n_{doc} = Jumlah total *tweet* dalam *data training*

2. Menghitung probabilitas (*likelihood)* masing-masing kata (a_i) untuk setiap kategori $P(a_i|v_j)$

$$P(a_i|v_j) = \frac{n_i}{kosa\ kata} \quad (1.6)$$

Dimana :

n_i = Jumlah kemunculan kata a_i dalam *tweet* yang berkategori v_j

kosa kata = Banyaknya kata dalam data *training*

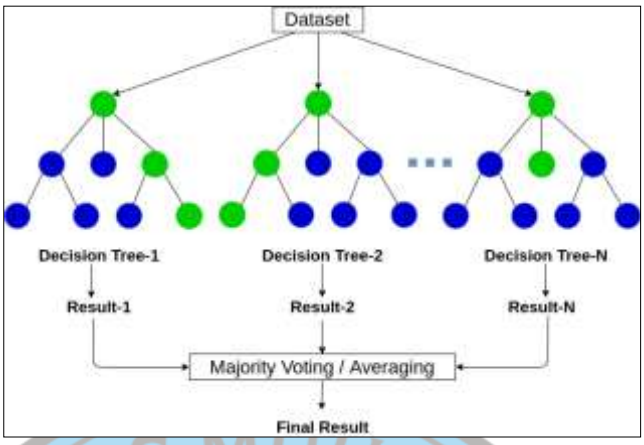
Masalah yang timbul dalam *training likeliood* adalah dimana kata yang dicari tidak terdapat dalam dokumen yang akan dihasilkan nilai 0. Untuk mengatasi hal ini maka dilakukan *smoothing* terhadap persamaan sebelumnya (2.6) maka persamaan akan menjadi :

$$P(a_i|v_j) = \frac{n_i+1}{n+kosa\ kata} \quad (1.7)$$

2.7 *Random Forest*

Random forest adalah salah satu metode *ensamble* (kumpulan) yang dikembangkan oleh Leo Breiman pada tahun 2001. *Random forest merupakan* metode pembelajaran mengunakan pohon keputusan (*decision tree)* sebagai *base classifier* dengan menerapkan metode *boostrap aggregating (bagging)*, dan *random feature selection.*

Random forest terdiri dari banyak *decision tree* yang banyaknya dimulai dari *tree*-1 sampai *tree-n,* dimana *n* adalah jumlah seluruh *tree* yang ada pada *random forest* tersebut. Tiap *tree* terdiri dari *root node* (node teratas) dan *leaf node* (hasil percabangan *root node).* Selanjutnya, *node* paling bawah akan menjadi penentu kelas.



Gambar 1.2 Algoritma *Random Forest*

Sumber : (faepa.br)

Algoritma dalam membangun *random forest* pada gugus data yang terdiri dari *n* amatan dan terdiri atas *p* peubah penjelas (prediktor), berikut tahapannya (Breiman dan Cutler, 2003):

1. Tahapan *Boostrapping*

   Tahahan *boostrapping* (*bagging)* melatih sejumlah pembelajaran dasar dari setiap sampel *boostrap* yang diperoleh dari subsampel yang sama dari kumpulan data *training.* Hal pertama yang dilakukan yaitu mengambil sampel secara acak berukuran *n* dari kumpulan data asli dengan pengembalian.

2. Tahapan *Random Feature Selection*

   Random forest dimulai dengan cara membentuk *tree* yaitu memanfaatkan algoritma *decision tree.* Langkah awalnya yaitu menghitung nilai *entropy* sebagai penentu tingkat kemurnian atribut dan nilai *information gain.* Menghitung nilai *entropy* target dapat menggunakan persamaan (2.8), untuk *entropy* variabel prediktor menggunakan persamaan (2.9) dan untuk

menentukan *information gain* menggunakan persamaan (2.10). Berikut persamaan yang digunakan :

$$Entropy\ (S) = \sum_{i=1}^{c} -p_i log_2 p_i \quad (1.8)$$

Keterangan:

S = Himpunan dataset

C = Jumlah kelas

p_i = Probabilitas frekuensi kelas ke-*i* dalam dataset

$$Entropy(prediktor) = \sum_{i=1}^{k} \frac{|S_i|}{|S|}\ x\ Entropy\ (S_i)$$

$$= \sum_{i=1}^{k} P(S_i)\ x\ Entropy\ (S_i) \quad (1.9)$$

Keterangan:

$P(S_i)$ = Peluang (S_i)

$|S_i|$ = Jumlah sampel untuk nilai *i*

$|S|$ = Jumlah seluruh sampel data

$Entropy\ (S_i)$ = *Entropy* untuk sampel yang memiliki nilai *i*

$$Gain\ (A) = Entropy\ (S) - Entropy\ (prediktor) \quad (1.10)$$

Keterangan:

S = Himpunan dataset

A = Atribut

Nilai *gain* yang tertinggi akan menjadi *root node.* Selanjutnya dihitung kembali *entropy* dan *gain* dari variabel yang tersisa hingga menghasilkan *entropy*=0 dimana merupakan akhir dari *node* (ujung percabangan).

Masing-masing *tree* akan menghasilkan hasil klasifikasi masing-masing. Hasil klasifikasi yang dihasilkan terbagi menjadi berbagai kelas sesuai yang telah ditentukan. Hasil voting terbanyak yang dihasilkan dari semua *tree* akan dijadikan *final-class*. *Final-class* ini akan dijadikan sebagai hasil klasifikasi dari algoritma *random forest* (Deshanta *et al,* 2020).

2.8 *Tuning* Parameter dengan *Grid SearchCV*

Meningkatnya jumlah data yang tersedia merupakan peluang untuk menggunakan informasi ini diberbagai jenis penelitian. Penelitian tersebut tidak memungkinkan upaya manusia tidak akan cukup untuk memproses data dalam jumlah yang besar. Oleh karena itu diperlukan adanya *machine learning* dalam menganalisis data dengan jumlah yang besar. *Machine Learning* merupakan pemrograman komputer untuk mewujudkan tugas yang diberikan menggunakan data yang dikumpulkan sebelumnya. Data masukan disebut data pelatihan (*train data)* yang disediakan secara manual oleh manusia atau dari kumpulan data yang besar dan terkadang kompleks (Guardiola, 2019).

Beberapa metode machine learning, terdapat nilai parameter yang diatur guna mendapatkan model yang optimal yang disebut *hyperparameter*. *Hyperparameter* (*tuning* parameter) digunakan untuk mengatur berbagai macam aspek dalam machine learning yang sangat berpengaruh pada performa dan model yang

dihasilkan. Salah satu metode *hyperparameter* yang dapat diaplikasikan adalah *grid search*. *Grid search* merupakan metode alternatif untuk mencari parameter terbaik untuk suatu model, sehingga algoritma pengklasifikasi dapat lebih akurat.

Grid search dikategorikan sebagai metode yang lengkap, karena parameter terbaik harus dilakukan uji coba atau dieksplorasi menggunakan setiap parameter yang ada. Menurut Ataei dan Osanloo (2004), setelah dilakukan eksplorasi, *grid search cross validation* akan menampilkan skor untuk setiap parameter, agar dapat dilakukan pengambilan skor terbaik dari parameter yang telah ditentukan.

Setiap metode klasifikasi mempunyai parameter yang berbeda dalam melakukan *hyperparameter tunning*. Seperti pada algoritma *naïve bayes* parameter yang digunakan untuk melakukan *hyperparameter* yaitu:

1. *Var_smoothing* digunakan untuk melakukan *smoothing* dalam proses *training likeliood*. *Variance* adalah bagian dari varians terbesar dari semua fitur yang ditambahkan ke varians untuk stabilitas perhitungan.
2. *Priors*. sedangkan *priors* merupakan probabilitas sebelumya dari kelas. *Priors* merupakan probabilitas sebelumnya dari kelas. Jika ditentukan, prior tidak disesuaikan dengan data.

Sedangkan pada metode klasifikasi *random forest* parameter yang digunakan yaitu:

1. *Max Depth* (digunakan untuk mengatur kedalaman maksimum pada *tree*).
2. *Criterion (*digunakan untuk mengukur kualitas split. Kriteria yang didukung adalah *"gini"* untuk ketidakmurnian gini *dan "entropy"* untuk perolehan informasi*).*

3. *Minimal Samples Split (*parameter untuk membentuk jumlah pengamatan minimum/pemisahan atau penentu berapa kali node splitting yang diperlukan pada simpul yang diberikan untuk membagi hutan acak, nilai *default* parameter ini adalah 2).
4. *Max Features* (Jumlah fitur yang dipertimbangkan saat mencari pemisahan terbaik, nilainya berupa 'sqrt', 'log2', dan *default*-nya adalah 'None'*).*

2.9 Evaluasi Model

Model dalam klasifikasi mempunyai arti dimana suatu model yang menerima masukan, kemudian mampu melakukan pemikiran terhadap masukan tersebut, dan dapat memberikan jawaban sebagai keluaran dari hasil pemikirannya. Sebuah model yang baik dapat melakukan klasifikasi semua set data dengan benar, tetapi tidak dapat dipungkiri bahwa kinerja juga harus diukur. Umumnya, pengukuran kinerja klasifikasi dilakukan dengan matriks konfusi (*confusion matrix).* Kinerja sistem klasifikasi menggambarkan seberapa baik sistem dalam mengklasifikasikan data. Sedangkan *confusion matrix* merupakan salah satu metode yang dapat digunakan untuk mengukur kinerja suatu metode klasifikasi (Roifa, 2018).

Dalam pengukuran *confusion matrix* terdapat 4 istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). Nilai TN merupakan jumlah data negatif yang terdeteksi benar, sedangkan FP merupakan data negatif namun terdeteksi sebagai data positif. Sementara itu, TP merupakan data positif yang terdeteksi benar dan FN merupakan data positif namun terdeteksi sebagai data negatif. *Confusion matrix* dapat disajikan seperti tabel berikut :

Tabel 1.9 Confusion Matrix

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

TP = True Positive, yaitu jumlah data positif yang terklasifikasi dengan benar oleh sistem.

TN =True Negative, yaitu jumlah data negatif yang terklasifikasi dengan benar oleh sistem.

FN =False Negative, yaitu jumlah data yang diprediksi negatif tetapi ternyata positif .

FP = False Positive, yaitu jumlah data yang diprediksi positif tetapi sebenarnya negatif.

Pada dasarnya *confusion matrix* mengandung informasi yang membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang seharusnya. Pengukuran efektif dapat dilakukan dengan perhitungan perolehan atau *recall*, nilai ketepatan atau presisi, nilai akurasi, dan nilai *F-1 score.* Membandingkan beberapa nilai tersebut terdapat beberapa acuan, yaitu sebagai berikut :

- Penggunaan akurasi sangat bagus digunakan sebagai acuan performasi algoritma jika data set memiliki jumlah data *False Negatif* dan *False Positif*

yang sangat mendekati (*simmetric).* Namun jika jumlahnya tidak mendekati, maka sebaiknya menggunakan *F-1 Score* sebagai acuan.

- Menggunakan *recall* tertinggi jika lebih memilih *False Positif* lebih baik terjadi daripada *False Negatif.* Dalam Dalam contoh, peneliti mempertimbangkan *recall* karena lebih baik algoritma memprediksi mahasiswa positif DO tetapi sebenarnya tidak DO daripada algoritma salah memprediksi bahwa mahasiswa diprediksi tidak DO padahal sebenarnya dia DO.
- Memilih presisi tertinggi jika menginginkan terjadinya *False Negatif* dan sangat tidak menginginkan terjadinya *False Positif.* Contohnya adalah pada kasus klasifikasi email SPAM atau tidak. Peneliti lebih memilih jika email yang sebenarnya SPAM namun diprediksi tidak SPAM (sehingga tetap ada pada kotak masuk email kita), daripada email yang sebenarnya bukan SPAM tapi diprediksi SPAM (sehingga tidak ada pada kotak masuk).

Berdasarkan nilai *True Negative* (TN), *False Positive* (FP), *False Negative* (FN), dan *True Positive* (TP) dapat diperoleh nilai akurasi, presisi, *recall*, dan nilai *F-1 score*. Nilai akurasi menggambarkan seberapa akurat sistem dapat mengklasifikasikan data secara benar. Nilai akurasi dapat diperoleh dengan berikut:

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN} \quad (1.11)$$

Kemudian, presisi merupakan proporsi jumlah dokumen yang ditemukan dan dianggap relevan untuk kebutuhan suatu informasi. Presisi dapat diperoleh dengan persamaan berikut :

$$Presisi = \frac{TP}{TP+FP} \quad (1.12)$$

Sementara itu, *recall* merupakan proporsi jumlah yang dapat ditemukan kembali dalam proses pencarian. Nilai recall diperoleh dengan persamaan berikut:

$$Recall = \frac{TP}{TP+FN} \quad (1.13)$$

F1-Score adalah *mean* dari presisi dan *recall*. *F1-Score* dapat diperoleh dengan persamaan berikut :

$$F-1\ score = \frac{2\ x\ (recall\ x\ presisi)}{(recall+presisi)} \quad (1.14)$$