BAB II
LANDASAN TEORI

2.1 Tinjauan Pustaka

2.1.1 Analisis Sentimen

Analisis sentimen atau *mining* opini merupakan bidang studi yang menganalisis opini, penilaian, dan emosi masyarakat terhadap sebuah produk, organisasi, individu, kejadian ataupun topik. Analisis sentimen berfokus pada opini yang mengekspresikan sentimen positif atau negatif [3]. Analisis sentimen digunakan untuk menemukan informasi berharga yang dibutuhkan dari data yang tidak terstruktur [12].

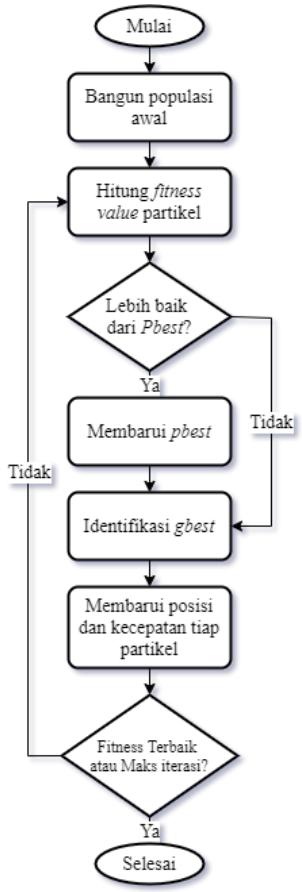
2.1.2 *Text Preprocessing*

Preprocessing merupakan tahapan persiapan dataset sebelum dilakukan pembobotan dan pemodelan yang bertujuan untuk mempermudah proses pengolahan data dan meningkatkan kualitas teks dengan melakukan beberapa proses yaitu [6]:

1. *Cleansing*, yaitu proses membersihkan data dari noise seperti *hashtag*, *username*, url, dan tanda baca.
2. *Case folding*, yaitu proses mengkonversikan teks dalam dokumen menjadi standar yang konsisten dalam huruf kecil.
3. *Stopword*, yaitu proses menghapus kata umum yang sering muncul dan tidak relevan seperti kata depan, kata ganti, kata sambung, dan sebagainya.
4. *Stemming*, yaitu proses merubah kata yang ada pada kalimat menjadi kata dasar.
5. *Tokenize*, yaitu proses memecah kalimat menjadi kata-kata yang lebih kecil atau token.

2.1.3 Particle Swarm Optimization

Particle Swarm Optimization merupakan algoritma optimasi yang diperkenalkan oleh James Kennedy dan Russel Eberhart yang terinspirasi dari perilaku kawanan burung dan ikan dalam bergerak untuk menghindari predator, mencari makanan dan pasangan dengan mengoptimalkan parameter lingkungan. Dengan mengamati dan mensimulasikan tingkah laku burung dan ikan agar dapat memodelkan tingkah laku manusia yang tentunya sangat berbeda dengan binatang dimana salah satu perbedaan yang penting ialah keabstrakan perilaku manusia yang tidak hanya disesuaikan melalui pergerakan fisik pada kegiatan sosial namun termasuk juga fungsi kognitif individu seperti keadaan mental, perasaan ataupun pengalaman seseorang dalam menghadapi masalah [14].

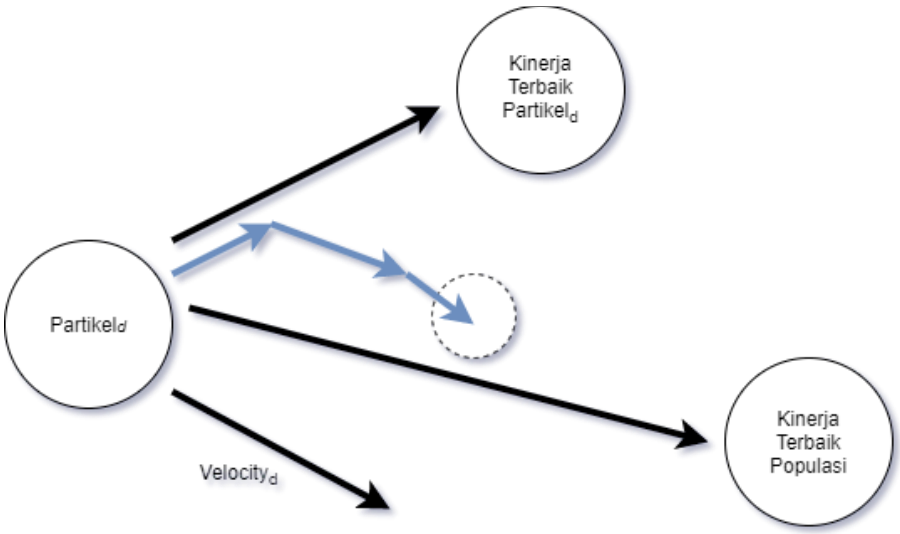


Gambar 2.1 *Flowchart* algoritma Particle Swarm Optimization [15]

Sebelum membentuk populasi awal, PSO perlu mengetahui ukuran populasi terlebih dahulu. Ukuran Populasi pada PSO bersifat tetap dimana semakin banyak

particle maka semakin cepat juga pencarian pada iterasinya namun membutuhkan waktu untuk mengevaluasi *particle* tiap iterasi dan berlaku juga sebaliknya. Secara empiris, ukuran populasi sebesar 20 sampai 30 *particle* sudah mencukupi dalam menyelesaikan masalah klasik. Inisialisasi PSO dimulai dengan mendistribusikan posisi dan *velocity particle* secara acak dan merata pada dimensi *D* ruang pencarian. Tiap iterasinya, PSO akan melakukan evaluasi berdasarkan *fitness value* dan akan melakukan perpindahan posisi *particle* untuk mencari *best fitness*. Pada ruang pencarian dimensi *D*, posisi saat ini *particle* pada iterasi *t* dapat diketahui melalui vektor *x(t)* dan *velocity*-nya ialah *v(t)*. Posisi kinerja terbaik *particle* (*pbest*) dapat diketahui pada *p(t)* dan posisi kinerja terbaik populasi (*gbest*) diketahui pada vektor *g(t)*

Pada dasarnya perpindahan *particle* dipengaruhi oleh 3 hal yaitu *velocity* dan kinerja terbaik *particle* tersebut, serta kinerja terbaik dalam populasi [16]. Binary Particle Swarm Optimization (BPSO) merupakan variasi PSO dimana tiap posisi *particle* pada *swarm* direpresentasikan berupa vektor bit biner 0 atau 1 dengan ukuran dimensi pencarian. Berbeda dengan Continuous PSO, pada BPSO *velocity* digunakan untuk menentukan apakah x_d seharusnya bernilai 0 atau 1 dimana *d* merupakan komponen pada *particle* x_t [17].



Gambar 2.2 Perpindahan *particle* pada PSO

Berdasarkan Gambar 2.2, perpindahan posisi *particle* dapat dilakukan dengan membandingkan nilai yang dihasilkan secara random *r* dengan *logistic funtion S*(v_d) pada persamaan (2.2) [17]. Logistic function dapat dihitung menggunakan persamaan (2.3) dengan nilai *velocity* yang dihasilkan melalui persamaan (2.1).

$$v_d = \omega v_d + rand(0, \varphi_1)(p_d - x_d) + rand(0, \varphi_2)(g_d - x_d) \quad (2.1)$$

$$x_d = \begin{Bmatrix} 1, rand < S(v_d) \\ 0, sebaliknya \end{Bmatrix} \quad (2.2)$$

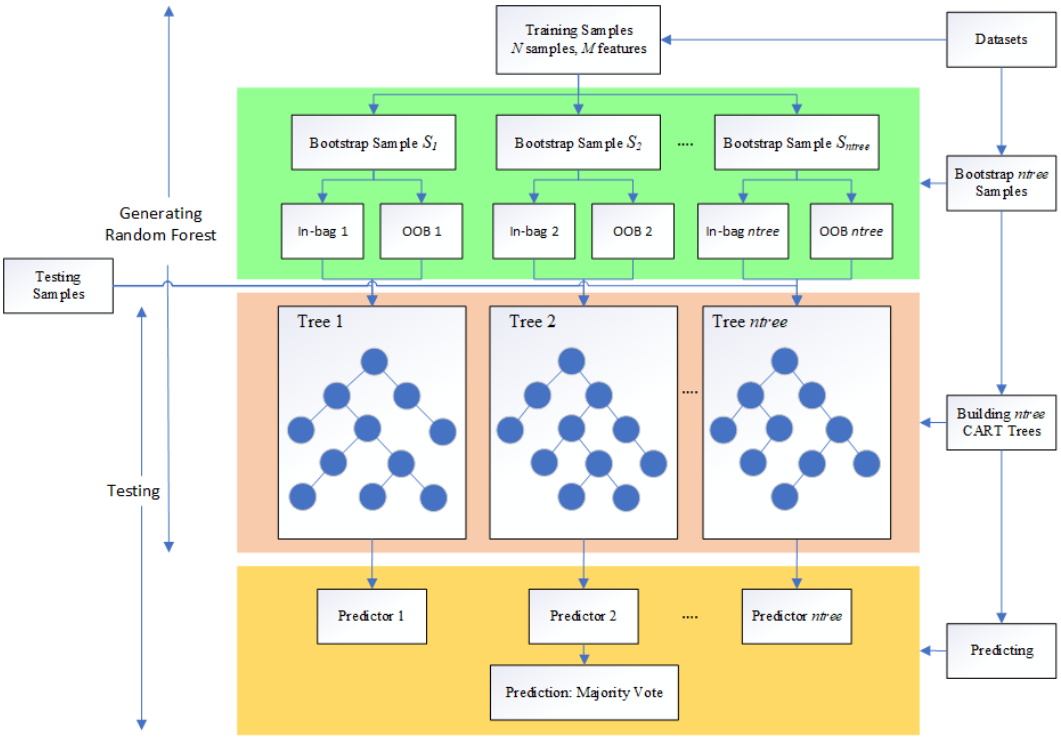
dimana,

$$S(v_d) = \frac{1}{1 + e^{-v_d}} \quad (2.3)$$

Melalui koefisien akselerasi $\varphi_1 \, dan \, \varphi_2$, pergerakan *particle* dapat lebih atau kurang responsif dan bahkan kemungkinan tidak stabil dengan peningkatan *velocity particle* tanpa kendali menyebabkan berbahaya terhadap pencarian. *Inertia Weight* (ω) merupakan parameter dalam PSO yang digunakan sebagai pengendali *velocity* pada lingkup pencarian. Bobot *Inertia* diusulkan untuk membatasi *velocity particle* dengan tujuan pergerakan *particle* dapat konvergensi dan lebih stabil. Nilai Inertia sebaiknya tidak bernilai terlalu kecil (misal 0,4) dikarenakan sistem menjadi lebih eksploitatif yang menyebabkan konvergensi prematur sedangkan nilai relatif besar (misal 0,9), pergerakan eksplorasi *particle* menjadi lebih luas menyebabkan konvergensi yang lambat [18].

2.1.4 Random Forest

Pada tahun 2001, Leo Breiman mengusulkan variasi *Bootstrap Aggregating* (*Bagging*) yang disebut dengan Random Forest (RF). Random Forest merupakan metode pembelajaran *ensemble* yang menggunakan Decision Tree sebagai dasar pengklasifikasi. Kumpulan *tree* tersebut tumbuh sesuai dengan *random vector* Θ_k, dimana $k = 1, 2, \dots, L$ yang didistribusi secara independen dan identik.



Gambar 2.3 *Flowchart* algoritma Random Forest [19]

Pembentukan Random Forest pada tiap iterasi $k = 1, 2, \ldots, L$ ialah dimulai dari proses *Bagging* yang merupakan pemilihan *sample*, S_k secara acak dari data *training D* dengan *n samples* dan *m* fitur asli. Setelah pemilihan *bootstrap sample*, *Tree* akan tumbuh dengan metode CART sampai ukuran maksimum dan tidak dipangkas. *Tree* akan memilih secara acak kandidat fitur *f*, dimana *f* lebih kecil daripada fitur asli *m*. Dari kandidat fitur *f* pilih fitur terbaik untuk memisah simpul dari kandidat [20]. Nilai *f* yang direkomendasikan ialah $\log_2 m + 1$. Pembentukan Random Forest dengan *Random Input Selection* ini dikenal juga sebutan Forest-RI [21].

Proses *Bagging* memungkinkan beberapa *sample* untuk dipilih lebih dari sekali ketika *sample* lain yang tidak akan dipilih. Sekitar 2 per 3 sampel digunakan sebagai data *training* model (disebut dengan *in-bag* sampel) dan sisa 1 per 3 digunakan sebagai validasi internal (disebut dengan *out-of-bag* sampel) [22]. Proses validasi internal ini dikenal *out-of-bag* (OOB) *error estimate*. Tiap data pengujian akan dievaluasi terhadap setiap *tree* yang memberikan suara untuk kelas data dimana

kelas data akan dipilih berdasarkan suara terbanyak atau yang disebut dengan *majority voting* [22].

2.1.5 *Confusion Matrix*

Confusion matrix merupakan alat berupa tabel yang berguna untuk mengevaluasi seberapa baik model mengenali data dengan kelas yang berbeda. Pada *confusion matrix* digunakan beberapa istilah yaitu:

1. *True Positive* (TP), merupakan jumlah data positif yang diprediksi dengan benar oleh model.
2. *True Negative* (TN), merupakan jumlah data negatif yang diprediksi dengan benar oleh model.
3. *False Positive* (FP), merupakan jumlah data negatif yang salah diprediksi sebagai positif oleh model.
4. *False Negative* (TN) merupakan jumlah data positif yang salah diprediksi sebagai negatif oleh model.

Atau dengan kata lain, TP dan TN memberitahukan ketika model melakukan sesuatu dengan benar, sementara FP dan FN memberitahukan ketika model melakukan kesalahan misalnya *mislabelling*. Jika terdapat kelas *z* dimana $z \geq 2$, maka ukuran *confusion matrix* setidaknya *z* * *z*. Berdasarkan *confusion matrix* dapat dilakukan pencarian nilai akurasi (*accuracy*) menggunakan persamaan (2.1) [21].

Tabel 2.1 *Confusion matrix class* z = 2

		Predicted Class		Total
		Positive	Negative	
Actual Class	Positive	TP	FN	P
	Negative	FP	TN	N
Total		P’	N’	P+N

$$Accuracy = \frac{TP+TN}{P+N} \quad (2.1)$$

2.2 Tinjuan Studi

Pada penelitian *An Integration of PSO-based Feature Selection and Random Forest for Anomaly Detection in IoT Network* pada tahun 2018, menjelaskan bahwa telah dilakukan penelitian klasifikasi pendeteksian *anomaly* pada keamanan jaringan IoT menggunakan metode usulan Random Forest berbasis *feature selection* PSO dan membandingkan model dengan metode Rotation Forest dan Deep Netral Network. Metode usulan kombinasi Random Forest dan PSO menghasilkan nilai performa tertinggi dengan nilai akurasi (A) sebesar 99,6%, Precision (P) sebesar 99,6%, Recall (R) sebesar 99,5% dan False Alarm Rate (FAR) sebesar 00,33% [23].

Pada penelitian *Feature Selection Based on Genetic Algorithm, Particle Swarm Optimization, Principal Component Analysis for Opinion Mining Cosmetic Product Review* pada tahun 2017, menjelaskan bahwa telah dilakukan penelitian evaluasi sentimen pada produk kosmetik Amazon.com menggunakan metode klasifikasi Support Vector Machine membandingkan algoritma *feature selection* PSO, GA dan GA + PCA. Kombinasi metode SVM dan PSO menghasilkan nilai performa tertinggi dengan nilai akurasi sebesar 97,00% dan Area Under Curve (AUC) sebesar 98,80% [2].

Pada penelitian Decision Tree Combined with PSO-Based Feature Selection for Sentiment Analysis pada tahun 2019, menjelaskan bahwa telah dilakukan penelitian evaluasi sentimen pada ulasan transportasi on-line di Indonesia menggunakan metode Decision Tree yang dikombinasikan dengan *Feature Selection* PSO menghasilkan peningkatan nilai akurasi dengan rata-rata sebesar 2,52% dan tertinggi 5,26% [7].

Pada penelitian Analisis Sentimen Aplikasi Ruang Guru di Twitter menggunakan Algoritma Klasifikasi pada tahun 2020, menjelaskan bahwa telah dilakukan penelitian evaluasi sentimen pada aplikasi Ruang Guru berbasis feature selection PSO menggunakan metode SVM, NB dan KNN. Penelitian ini menghasilkan nilai performa tertinggi pada kombinasi SVM dan PSO dengan nilai akurasi sebesar 78,55% dan AUC sebesar 85,30% [12].

Pada penelitian *Random Forest Approch for Sentiment Analysis in Indonesia Language* pada tahun 2018, menjelaskan bahwa telah dilakukan penelitian evaluasi sentimen *review* dari FemaleDaily menggunakan metode Random Forest. Penelitian ini menghasilkan nilai rata-rata OOB score sebesar 82,90% [24].

Tabel 2.2 Penelitian terkait

Judul	Classifier	Hasil	Keterangan
An Integration of PSO-based Feature Selection and Random Forest for Anomaly Detection in IoT Network (2018) [23]	Rotation Forest (RoF) + PSO	A: 99,40% P: 97,60% R: 97,90% FAR: 0,49%	Masalah pada penelitian ini ialah mendeteksi gangguan *Anomaly* pada jaringan IoT dengan klasifikasi kelas Normal dan *Malicious*.
	Deep Neural Network (DNN) + PSO	A: 99,30% P: 99,40% R: 99,10% FAR: 0,54%)	
	RF + PSO	**A: 99,60%** **P: 99,60%** **R: 99,50%** **FAR: 0,33%**	
Feature Selection Based on Genetic Algorithm, Particle Swarm Optimization, Principal Component Analysis for Opinion Mining Cosmetic Product Review (2017) [2]	SVM	A: 82,00% AUC: 98,80%	Dataset : Review Kosmetik Amazon.com. Tools: RapidMiner. Preprocess : Tokenize, Generate N-Gram, Stemming. Feature Weighting : TF-IDF.
	SVM + PSO	**A: 97,00%** **AUC: 98,80%**	
	SVM + GA	A: 94,00% AUC: 98,40%	
	SVM + GA + PCA	A: 83,00% AUC: 80,90%	

Judul	Classifier	Hasil	Keterangan
Decision Tree Combined with PSO-Based Feature Selection for Sentiment Analysis (2019) [7]	**Decision Tree (C4.5) + PSO**	**Peningkatan Akurasi rata-rata sebesar 2,52% dengan nilai tertinggi 5,26%**	Dataset: Review on-line transportation in Indonesia. Tools: Python Scikit-Learn. Preprocess: Case folding, Tokenizing, Stemming, dan Stopword Removal. Feature Weighting: TF-IDF.
Analisis Sentimen Aplikasi Ruang Guru di Twitter Menggunakan Algoritma Klasifikasi (2020) [12]	**SVM + PSO**	**A: 78,55%** **AUC : 85,30%**	Dataset: Tweet Ruang Guru. Tools: RapidMiner. Preprocess: Transform case, Remove Http dan @, Tokenize, Filter Token, Stopword Removal, SMOTE Upsampling.
	NB + PSO	A: 67,32% AUC : 46,30%	
	KNN + PSO	A: 77,21% AUC : 83,10%	
Random Forest Approch for Sentiment Analysis in Indonesia Language (2018) [24]	**Random Forest**	**OOB score: 85,90%**	Dataset: *Review* dari FemaleDaily. Preprocess: Tokenization, Case Folding dan Cleaning. Feature Extraction: Binary TF, Raw TF, Log TF, dan TF.IDF.