BAB II

TINJAUAN PUSTAKA

2.1 Analisis Sentimen

Analisis sentimen adalah suatu bidang dari *Natural Language Processing* (NLP) yang merupakan proses identifikasi kasus yang memiliki *dataset* berupa teks berdasarkan pandangan terhadap suatu aspek. Pada umumnya analisis sentimen dilakukan implementasi pada berbagai aspek, seperti sentimen seputar berita, layanan dan objek lain (Waluyan dan Hartomo 2022).

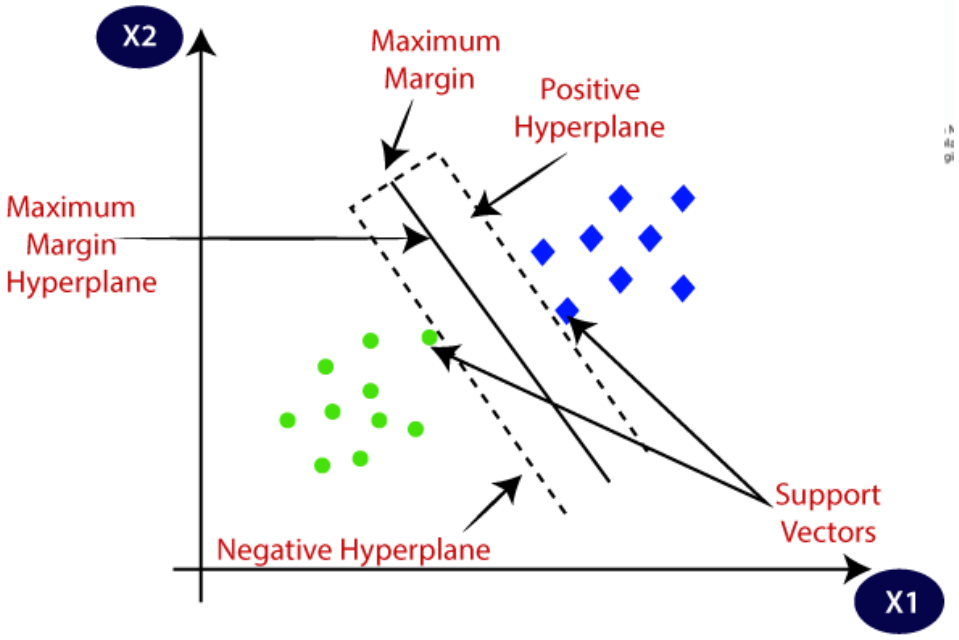
Analisis sentimen telah diterapkan secara luas di berbagai bidang, mulai dari sektor pariwisata, *e-commerce*, transportasi, hiburan, hingga layanan telekomunikasi. Penerapan analisis ini tidak hanya terbatas pada satu jenis aplikasi tertentu, tetapi juga memberikan kontribusi yang signifikan dalam mendukung pengambilan keputusan strategis di berbagai sektor. Dengan menganalisis opini atau ulasan dari pengguna, analisis sentimen mampu membantu perusahaan dan organisasi dalam memahami kebutuhan, preferensi, dan kepuasan pelanggan, sehingga dapat digunakan untuk meningkatkan kualitas layanan atau produk yang ditawarkan (Nufairi, Pratiwi, dan Herlando 2024).

2.2 *Support Vector Machine* (SVM)

Support Vector Machine (SVM) merupakan salah satu metode yang bisa digunakan untuk pengklasifikasian (Samantri dan Afiyati 2024). Banyak penelitian sebelumnya yang menggunakan metode ini sebagai metode untuk mengklasifikasi kasus analisis sentimen (Najib dkk. 2019).

SVM digunakan sebagai algoritma klasifikasi utama dalam penelitian ini untuk menentukan sentimen dari teks yang telah diproses. *SVM* dipilih karena kemampuannya dalam menangani data berdimensi tinggi, seperti fitur berbasis teks yang dihasilkan dari metode *Term Frequency-Inverse Document Frequency* (TF-IDF) atau *word embeddings*.

Klasifikasi dengan metode *SVM* yang dilakukan pertama kali ialah mencari *hyperplane* atau garis pembatas yang memisahkan antara suatu kelas dengan kelas lain (Permatasari, Linawati, dan Jasa 2021). Untuk dapat menemukan *hyperplane* terbaik, harus diberikan maksimal margin yang merupakan salah satu ciri dari metode *SVM* (Tri Putra, Amin Hariyadi, dan Crysdian 2023). Ilustrasi dari metode *SVM* ditujukkan pada Gambar 2.1.



Gambar 2.1 Ilustrasi *Support Vector Machine* (Edeib dkk. 2023)

Berdasarkan Gambar 2.1, persamaan untuk menghitung proses perhitungan *SVM* dapat dilihat pada persamaan 2.1 dan persamaan 2.2:

$$h(x) = w.x + b \tag{2.1}$$

Keterangan:

w = vektor yang tegak lurus dengan *hyperplane*

x = data

b = nilai bias

$h(x)$ = fungsi *hyperplane*

$$Margin = |d_{h_1} - d_{h_2}| = \frac{2}{||w||} \tag{2.2}$$

Keterangan:

d_{h_1}= jarak *hyperplane* kelas +1

d_{h_2}= jarak *hyperplane* kelas -1

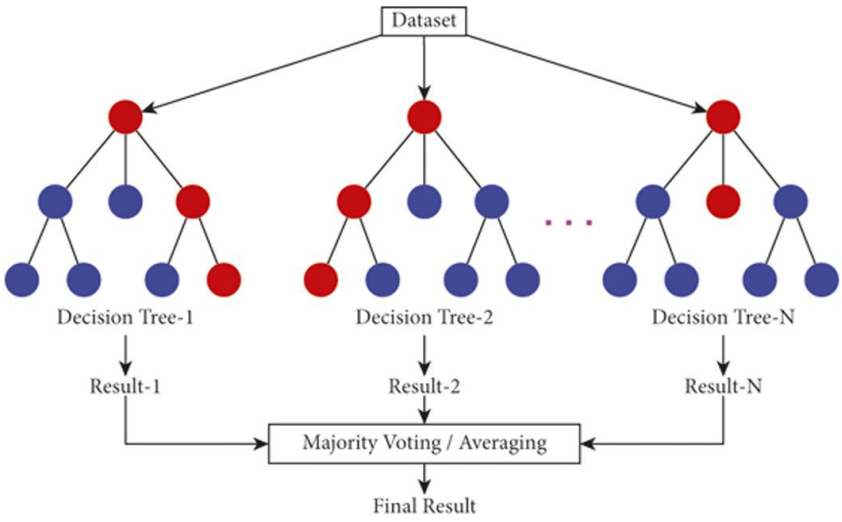
2.3 *Random Forest*

Random Forest adalah algoritma yang menggunakan metode pembagian biner rekursif untuk mencapai node terakhir dalam struktur pohon berdasarkan pohon klasifikasi (Azizan, Andini, dan Azzahra 2023). *Random Forest* pertama kali dikembangkan oleh L. Breiman pada tahun 2001. Algoritma ini biasanya digunakan untuk melakukan klasifikasi data dan sebagai metode regresi.

Random Forest digunakan dalam penelitian ini untuk mengevaluasi pentingnya fitur dalam menentukan sentimen teks. Dengan menggunakan kombinasi banyak pohon keputusan *(decision trees)*, *Random Forest* memberikan

estimasi terhadap kontribusi setiap kata dalam prediksi sentimen, yang dapat digunakan untuk memahami pola kata yang paling berpengaruh.

Random Forest mampu menghasilkan beberapa pohon independen dengan subset yang dipilih secara acak melalui bootstrap dari sampel pelatihan dan variabel masukan di setiap node (N. B. Putri dan Wijayanto 2022). Seperti namanya "*forest*" yang melambangkan kumpulan banyak pohon (Sidauruk, Riza, dan Siti Fatonah 2023). Algoritma ini menggabungkan berbagai pohon keputusan (decision trees) untuk membentuk satu model yang lebih efektif secara keseluruhan (Di dkk. 2022). Klasifikasi dalam *Random Forest* ditentukan berdasarkan sistem voting dari setiap pohon yang telah terbentuk. Pohon dengan suara terbanyak akan menentukan hasil klasifikasi yang akan diambil (Afdhal dkk. 2022). Ilustrasi dari metode *Random Forest* ditujukkan pada Gambar 2.2.



Gambar 2.2 Visualisasi *Random Forest* (Harisdianto dkk. 2023)

Secara garis besar, algoritma *Random Forest* beroperasi melalui tahapan tahapan berikut.

1. Dari kumpulan data latih (*training set*), metode random repeated sampling diaplikasikan, lalu sejumlah K pohon dibangun untuk regresi dan klasifikasi.
2. Sejumlah m fitur dipilih secara acak dari total n fitur dalam *training* set, dengan m kurang dari atau sama dengan n. Dengan mengkalkulasi informasi yang terkandung dalam masing-masing dari m fitur tersebut, fitur yang paling berpengaruh dalam penentuan klasifikasi akan dipilih untuk melakukan pemisahan node (*node splitting*).
3. Setiap pohon akan terus berkembang hingga mencapai ukuran maksimal atau hingga batas yang telah ditetapkan sebelumnya.

Menggunakan semua pohon yang telah terbentuk, sampel data baru dapat diklasifikasikan berdasarkan tiap-tiap pohon. Hasil klasifikasi akhir ditentukan berdasarkan jumlah suara (*vote*) yang diperoleh dari masing-masing pohon tersebut. Persamaan untuk menghitung proses *Random Forest* dapat dilihat pada persamaan 2.3:

$$\hat{C}_{rf}^{B}(x) = majorityvote\left(\hat{C}_b(x)\right)_1^B \quad (2.3)$$

Keterangan:

$\hat{C}_{rf}^{B}(x)$ = Kelas prediksi dari pohon *Random Forest* ke-b

2.4 *Extreme Gradient Boosting (XGBoost)*

Extreme Gradient Boosting (*XGBoost*) adalah metode *Machine Learning* yang digunakan untuk menyelesaikan masalah klasifikasi dan regresi (Dava Maulana dkk. 2023). Algoritma ini menerapkan teknik *Gradient Boosting Decision Tree* (GBDT), beberapa model kecil biasanya berupa *Decision Trees* digabungkan

untuk membentuk model yang lebih kuat dan memberikan hasil akurasi yang lebih baik. Pohon keputusan dalam *XGBoost* dibangun secara sekuensial, dengan kedalaman yang seragam, sehingga prinsip utama yang digunakan adalah *depth-wise* atau *level-wise* (Arif Farid 2023). *XGBoost* juga memperkenalkan berbagai teknik tambahan untuk meningkatkan efisiensi dan efektivitasnya, seperti *regularization* dan modifikasi pada *loss function* yang membantu dalam mengurangi *overfitting* dan meningkatkan kemampuan prediksi model (Ikegami dan Darmawan 2022). Berikut merupakan cara pembuatan pohon *XGBoost*:

1. Menentukan nilai probabilitas awal dari data target, *class value* yaitu jumlah data yang akan diolah, dapat dilihat pada persamaan 2.4:

$$Probality(p) = \frac{\Sigma(Class\ Value)}{\Sigma(Class)} \tag{2.4}$$

2. Menentukan nilai *residual* dengan mengurangkan nilai *class* setiap data dengan nilai probabilitas awal, dapat dilihat pada persamaan 2.5:

$$Residual(Y) = Class\ Value - Probality \tag{2.5}$$

3. Membuat *root* awal dari *classification tree* dengan *residual* yang telah ditentukan dengan menjumlahkan semua *residual* tersebut. Selanjutnya, membuat *leaf* atau dengan mengklasifikasikan berdasarkan fitur yang ada.
4. Menghitung *similarity score* atau kesamaan antara data, dapat dilihat pada persamaan 2.6:

$$Similarity\ Score = \frac{(\Sigma Residual)^2}{\Sigma(p\ x\ (1\ -\ p)\ +\lambda\ )} \tag{2.6}$$

Keseluruhan nilai *residual* dimasukkan ke dalam satu *leaf* yang sama dan dihitung nilai *similarity score* dari *leaf*.

5. Setelah menghitung semua *similarity* dan setiap *leaf*, selanjutnya menghitung nilai *gain* dari *left similarity* dan *right similarity*, dapat dilihat pada persamaan 2.7:

$$Gain = \frac{1}{2}[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} + \frac{(G_L + G_R)^2}{(H_L + H_R) + \lambda}] - y \quad (2.7)$$

Dari semua kemungkinan nilai yang didapatkan setelah pemisahan setiap sampel yang diobservasi, nilai gain tertinggi yang dipilih menjadi cabang yang memisahkan *residual*.

6. Kemudian melakukan iterasi atau percabangan menggunakan pruning untuk menghitung selisih antara *gain* dari cabang paling bawah dari *tree* dengan nilai *gamma* yang sudah ditetapkan, dapat dilihat pada persamaan 2.8:

$$L(\theta) = \sum_{i}^{n} (y_i - y)^2 \quad (2.8)$$

Apabila dalam operasi kondisi tersebut mendapatkan hasil <0, maka *leaf* tersebut dipangkas dan data tersebut tidak digunakan lagi. Namun, jika bernilai >0 artinya leaf tidak dapat dipangkas.

7. Kemudian menghitung *input value* dari setiap *leaf*, dapat dilihat pada persamaan 2.9:

$$Similarity\ Score = \frac{\Sigma(Residual)}{\Sigma(p\ x\ (1 - p) + \lambda)} \quad (2.9)$$

8. Kemudian menghitung *input value* dari setiap *leaf*, dapat dilihat pada persamaan 2.10:

$$Scale\ data(P) = (\frac{p}{1} - p) + (learning\ rate\ x\ output\ value) \quad (2.10)$$

Nilai *learning rate* biasanya dengan range 0-1, dapat dilihat pada persamaan 11:

$$Probality\ baru = \frac{e^P}{1 + e^P} \quad (2.11)$$

Selanjutnya membuat probabilitas baru dengan menggabungkan nilai-nilai.

9. Mengulangi langkah-langkah dari nomor 4 dengan model *tree* dan kondisi fitur yang berbeda sehingga nilai residual seminimal mungkin atau sampai mendapatkan jumlah tree yang maksimal.

2.5 *Term Frequency-Inverse Document Frequency* (TF-IDF)

Term Frequency (TF) merujuk pada frekuensi kemunculan suatu term dalam sebuah dokumen. Nilai TF untuk setiap dokumen bisa berbeda-beda, bergantung pada tingkat kepentingan sebuah term dalam dokumen tersebut. Sementara itu, *Inverse Document Frequency* (IDF) mengukur sejauh mana suatu term tersebar di seluruh koleksi dokumen. Semakin sedikit dokumen yang mengandung term tertentu, semakin besar nilai IDF-nya (Kusnadi dkk. 2021). Jika sebuah term ada pada setiap dokumen dalam koleksi, maka nilai IDF untuk term tersebut adalah nol (Ardras dan Voutama 2023). Hal ini menunjukkan bahwa term yang muncul di semua dokumen tidak memberikan informasi yang cukup untuk membedakan dokumen berdasarkan topik tertentu. Berikut merupakan rumus dari TF-IDF:

1) *Term Frequency* (TF) adalah jumlah kemunculan kata dalam sebuah dokumen.
2) *Inverse Document Frequency* (IDF) adalah logaritma dari total jumlah dokumen dibagi jumlah dokumen yang mengandung kata tersebut.
3) TF-IDF untuk setiap kata dihitung dengan mengalikan TF dan IDF.
4) Menganalisis hasil TF-IDF untuk melihat kata-kata mana yang paling signifikan dalam komentar.

Informasi ini dapat digunakan untuk analisis sentimen, misalnya dengan mengidentifikasi kata-kata dengan sentimen positif atau negatif, dapat dilihat pada persamaan 2.12, persamaan 2.13 dan persamaan 2.14:

$$TF(t,d) = \frac{Jumlah\ kemunculan\ t\ dalam\ dokumen\ d}{Jumlah\ kata\ dalam\ dokumen\ d} \tag{2.12}$$

$$IDF(t) = log\frac{Total\ Jumlah\ Dokumen}{Jumlah\ dokumen\ yang\ mengandung\ kata\ t} \tag{2.13}$$

$$TF - IDF(t,d) = TF(t,d) \times IDF(t) \tag{2.14}$$

2.6 *Lexicon*

Lexicon merupakan metode yang digunakan untuk menentukan sentimen dengan menghitung orientasi dari suatu dokumen. Orientasi ini dihitung berdasarkan kata atau frasa yang terdapat dalam dokumen. Dalam pendekatan ini, kamus digunakan sebagai acuan untuk menentukan kelas sentimen suatu dokumen (Ismail dan Raden Bagus Fajriya Hakim 2023).

Kamus sentimen dapat dibentuk secara manual menggunakan daftar kata yang telah disusun sebelumnya. Sebagian besar penelitian berfokus pada penggunaan kata-kata orientasi yang telah ditentukan untuk menganalisis teks.

Kamus ini bisa juga dibentuk secara otomatis dengan pendekatan berbasis semantik. Proses analisis dimulai dengan menentukan daftar kata yang sesuai untuk penelitian. Dalam analisis kalimat, setiap kata sifat pada suatu kalimat diberi nilai dan nilai-nilai tersebut kemudian digabungkan untuk menghasilkan skor tunggal yang menentukan kelas sentimen suatu dokumen (Salim dan Mayary 2020).

2.7 *Feature Selection Chi-Square*

Feature Selection Chi-Square merupakan pemilihan kata untuk mereduksi fitur yang tidak relevan setelah proses TF-IDF. Seleksi Fitur dengan *Chi-Square* dilakukan dengan cara pengurutan pada tiap fitur berdasarkan bobot dari nilai terbesar hingga terkecil (Hokijuliandy, Napitupulu, dan Firdaniza 2023).

Chi-Square digunakan dalam penelitian ini untuk melakukan seleksi fitur dengan tujuan mengurangi dimensi data dan meningkatkan efisiensi model klasifikasi. Teknik ini mengevaluasi hubungan antara kata-kata dalam teks dengan label sentimen, sehingga hanya kata-kata yang memiliki hubungan signifikan yang dipertahankan sebagai fitur dalam model pembelajaran mesin. Dengan menerapkan seleksi fitur berbasis *Chi-Square*, kita dapat menghilangkan kata-kata yang kurang relevan, sehingga dapat meningkatkan akurasi model serta mempercepat waktu komputasi dalam proses pelatihan. Perhitungan nilai *Chi-Square* kepada tiap kata yang muncul pada kelas *c* dapat dibantu dengan menggunakan tabel kontingensi.

Tabel 2.1 Kontingensi *Chi-Square*

Kata	Kelas	
	C	c^2
t	A	B
t^2	C	D

Nilai pada Tabel 2.1 merupakan nilai frekuensi observasi dari suatu kata terhadap kelas dengan keterangan sebagai berikut:

A = banyaknya dokumen pada kelas *c* yang memuat kata *t*

B = banyaknya dokumen yang tidak ada di kelas *c* namun ada kata *t*

C = banyaknya dokumen yang ada di kelas *c* namun tidak memiliki kata *t*

D = banyaknya dokumen yang tidak ada di kelas *c* dan tidak memuat kata *t*

Pada tahap ini, tiap kata yang didapatkan nantinya dikalkulasi memakai persamaan 2.15:

$$X^2(t,c) = \frac{N(AD - CB)^2}{(A + C)(B + D)(A + B)(C + D)} \tag{2.15}$$

Setelah perhitungan *Chi-Square* dilakukan, fitur yang memiliki nilai *Chi-Square* lebih besar dari nilai kritis ditentukan pada tingkat signifikansi tertentu akan dipilih. Semakin kecil taraf α (tingkat signifikansi), semakin besar nilai kritis yang diperlukan untuk memilih fitur, sehingga hanya fitur-fitur yang memiliki hubungan yang signifikan dengan kategori tertentu yang akan dipertahankan. Hal ini akan mengurangi jumlah fitur yang digunakan dalam analisis, mengurangi beban data, dan meningkatkan efisiensi serta akurasi dalam proses klasifikasi (Harungguan, Napitupulu, dan Firdaniza 2023).

2.8 *Confusion Matrix*

Evaluasi model klasifikasi didasarkan pada pengukuran terhadap kinerja dari model klasifikasi untuk menggambarkan seberapa baik sistem dalam mengklasifikasikan data. *Confusion matrix* menyajikan informasi yang membandingkan hasil klasifikasi yang diberikan oleh sistem dengan hasil klasifikasi yang seharusnya. Metode ini umum digunakan untuk mengukur kinerja model klasifikasi. Setiap sel dalam *confusion matrix* menunjukkan jumlah kasus yang sebenarnya dari masing-masing kelas yang diamati dan bagaimana kelas tersebut diprediksi oleh model (Julianto 2022).

Confusion matrix biasanya disajikan dalam bentuk tabel seperti yang ditunjukkan pada Tabel 2.2. Dengan menggunakan informasi yang terdapat dalam *confusion matrix*, tingkat akurasi hasil klasifikasi dapat dihitung untuk mengevaluasi kinerja model.

Tabel 2.2 *Confusion matrix*

		Data Sebenarnya	
		POSITIVE	***NEGATIVE***
Hasil Prediksi	***POSITIVE***	*True Positive* (TP) *correct result*	*False Positive* (FP) *unexpected result/false alarm*
	NEGATIVE	*False Negative* (FN) *missing result*	*True Negative* (TN) *correct rejection*

1. *True Positive* (TP): kondisi dimana data yang sebenarnya bernilai positif diprediksi sebagai positif.
2. *False Positive* (FP): kondisi dimana data yang sebenarnya bernilai negatif diprediksi sebagai positif.
3. *True Negative* (TN): kondisi dimana data yang sebenarnya bernilai negatif diprediksi sebagai negatif.
4. *False Negative* (FN): kondisi dimana data yang sebenarnya bernilai positif diprediksi sebagai negatif.

2.9 Penelitian Terkait

Literatur yang dijadikan referensi dalam penelitian ini mencakup jurnal jurnal ilmiah nasional dan internasional yang berfokus pada analisis sentimen dengan menggunakan berbagai model maupun metode.

Tabel 2.3 Penelitian Terkait

Penulis	**Judul Penelitian**	**Metode**	**Hasil**
(Wahyudi dan Kusumawardana 2021)	Analisis Sentimen pada Aplikasi Grab di *Google Play Store* Menggunakan *Support Vector Machine*	*Support Vector Machine*	Hasil dari analisis menggunakan *Support Vector Machine* menghasilkan akurasi 85,54% dan Hasil Review positif yang paling sering diulas adalah ”ovo”, sedangkan review negatif yang paling sering diulas adalah “driver”.
(Gifari dkk. 2022)	Analisis Sentimen Review Film Menggunakan TF-IDF dan *Support Vector Machine*	*Support Vector Machine*	Dari uji skenario yang dilakukan, diketahui bahwa algoritma TF-IDF dan *SVM* dapat digunakan untuk kasus

Penulis	Judul Penelitian	Metode	Hasil
			review film dengan nilai *Accuracy* 85%, nilai *Precision* 100%, nilai *Recall* 70%, dan nilai *F1-Score* sebesar 82%.
(Tuhuteru 2020)	Analisis Sentimen Masyarakat Terhadap Pembatasan Sosial Berksala Besar Menggunakan Algoritma *Support Vector Machine*	*Support Vector Machine*	Hasil yang didapatkan yaitu nilai akurasi sebesar 82.07%, nilai *recall* sebesar 82.07%, dan nilai *precision* sebesar 83.20%.
(Ardianto dkk. 2020)	Sentiment Analysis On E-Sports For Education Curriculum Using Naïve Bayes And *Support Vector Machine*	Naïve Bayes dan *Support Vector Machine*	Algoritma Naïve Bayes dengan SMOTE mendapatkan nilai akurasi 70.32%, dan nilai AUC 0.954. Sedangkan *Support Vector Machine* dengan SMOTE mendapatkan nilai akurasi 66.92% dan nilai AUC 0.832. Perbedaan akurasi antara naïve bayes dengan *Support Vector Machine* 3.4%.
(Styawati dkk. 2022)	Sentiment Analysis on Online Transportation Reviews Using Word2Vec Text Embedding Model Feature Extraction and *Support Vector Machine* (*SVM*) Algorithm	*Support Vector Machine*	Nilai akurasi sebesar 89%, presisi sebesar 94%, *recall* 86% dan *F1-Score* 90%. Sedangkan aplikasi Grab memiliki nilai akurasi sebesar 87%, presisi sebesar 94%, *recall* sebesar 85%, dan *F1-Score* sebesar 89%.
(Yasir dkk. 2024)	Analisis Sentimen Terhadap Kontroversi Fatwa MUI Nomor 83 Tahun 2023 Tentang Pemboikotan Produk yang Terafiliasi Israel	Naïve Bayes, Decision Tree, *Random Forest, SVM, dan KNN*	Hasil analisis menunjukkan tiga klasifikasi sentimen: setuju, tidak setuju, dan netral, dengan tingkat akurasi Naive Bayes 75%, Decision Tree 65%, *Random Forest* 67%, *SVM* 63%, KNN 53%.
(Jayanti dkk. 2024)	Analisis Sentimen Penggunaan Aplikasi	*Random Forest, Support*	Hasilnya menunjukkan bahwa *SVM* memiliki

Penulis	Judul Penelitian	Metode	Hasil
	Traveloka di Twitter Menggunakan Model Klasifikasi	*Vector Machine (SVM), Naive Bayes Classifier, K-Nearest Neighbors (KNN),* dan *XGBOOST*	akurasi lebih baik berdasarkan evaluasi matriks dengan nilai sebesar 90%. Namun, melalui uji model menggunakan AUC, *XGBOOST* memperoleh nilai tertinggi sebesar 71%.
(Sondakh dkk. 2023)	Sistem Analisis Sentimen Ulasan Aplikasi Belanja Online Menggunakan Metode Ensemble Learning	*SVM, KNN,* dan *Random Forest*	Ensemble learning menghasilkan model classifier dengan performa yang lebih baik, dengan indikator akurasi 81.8% *precision* 83%, *recall* 82%, *F1-Score* 82%
(Ahmad Dzulhijjah dkk. 2023)	Perbandingan Metode *Random Forest* dan KNN pada Analisis Sentimen Twitter	*Support Vector Machine* dan *K-Nearest Neighbors*	Setelah dilakukan pengujian dan evaluasi didapatkan hasil akurasi dari algoritma *SVM* sebesar 83% dan KNN sebesar 49%.
(Rahmawati dan Sukmasetya 2022)	Sentimen Analisis Opini Masyarakat Terhadap Kebijakan Kominfo atas Pemblokiran Situs non-PSE pada Media Sosial Twitter	Decision Tree, *KNN,* Naïve Bayes, *Random Forest,* Logistic Regression dan *SVM*	Hasil dari penelitian ini menyatakan bahwa sebanyak 1234 tweet yang telah dilakukan pre-prosesing cenderung bermakna negatif dengan presentase sebesar 82.82%. sedangkan tweet positif bernilai 10.53% dan tweet netral sebesar 6.65%. Serta metode yang bernilai akurasi tinggi pada penelitian ini adalah metode KNN dan *Random Forest* dengan akurasi 85.8%
(Erkamim dkk. 2023)	Komparasi Algoritme *Random Forest* dan *XGBoost*ing dalam Klasifikasi Performa UMKM	*Random Forest* dan XGBoosting	*Random Forest*: Akurasi = 0,944 dan *F1-Score* = 0,944, *XGBoost*ing: akurasi 0,944 dan *F1-Score* 0,950.

Penulis	Judul Penelitian	Metode	Hasil
(Oktavia dan Ramadahan 2023)	Analisis Sentimen Terhadap Penerapan Sistem E-Tilang Pada Media Sosial Twitter Menggunakan Algoritma *Support Vector Machine* (*SVM*)	*Support Vector Machine* (*SVM*)	Akurasi = 74.20%, *precision* = 83.33% dan *recall* = 5.28%
(Mardiyati dkk. 2023)	Analisa Prediksi Tegangan *Input* Sensor Capcitive Soil Moisture dengan *Random Forest* untuk Mendukung Pertanian Pintar	*Random Forest*	Akurasi = 100%, *precision* = 1.00, *recall* = 1.00, *F1-Score* = 1.00, *support* = 45, MSE dan RMSE = 0,0000
(Kurniawan dan Indahyanti 2024)	Prediksi Angka Harapan Hidup Penduduk Menggunakan Metode *XGBoost*	*XGBoost*	Akurasi = 96,8%, MAE = 0,97.
(Azizan, Andini, dan Azzahra 2023)	Analisis Perbandingan Metode Klasifikasi Menggunakan Regresi Logistik Biner, *Random Forest*, dan *Support Vector Machine* pada Cardiovascular Disease *Dataset*	Regresi Logistik Biner, *Random Forest*, dan *Support Vector Machine*	*Random Forest*: Akurasi = 73.2%, *recall* = 66,7%, AUC = 80,3%. *Support Vector Machine* (*SVM*): Spesifisitas = 82,5%
(N. B. Putri dan Wijayanto 2022)	Algoritma *XGBoost* Untuk Klasifikasi Kualitas Air Minum	*XGBoost*	Akurasi = 82,29%, *precision* = 78,62%, *recall* = 85.90% dan *F1-Score* = 82.09%.
(Topaloglu 2024)	A hybrid approach based on k-means and *SVM* algorithms in selection of appropriate risk assessment methods for sectors	*Support Vector Machine*, *Feature Selection Chi-Square*	Akurasi hybrid method = 96,63%, Akurasi *SVM* = 94,68%
(D. J. Putri, Dwifebri, dan Adiwijaya 2023)	Text Classification of Indonesian Translated Hadith Using *XGBoost* Model and *Chi-square* Feature Selection	*XGBoost, Feature Selection Chi-Square*	Akurasi = 75%, *precision* = 75%, *recall* = 74%, *F1-Score* = 74%
(Tasnim dan Habiba 2021)	A Comparative Study on Heart Disease Prediction Using Data	*Random Forest*, *XGBoost*,	Akurasi *XGBoost* = 83.5%, Akurasi *Random*

Penulis	Judul Penelitian	Metode	Hasil
	Mining Techniques and Feature Selection	*SVM*, *Feature Selection Chi-Square*	*Forest* = 92.85%, Akurasi *SVM* = 82%
(Ratiasasadara, Sudarno, dan Tarno 2023)	Analisis Sentimen Penerapan PPKM Pada *Twitter* Menggunakan *Naïve Bayes Classifier* Dengan Seleksi Fitur *Chi-Square*	*Naïve Bayes, Feature Selection Chi-Square*	Akurasi Naïve Bayes = 83%
(Septiana dan Alita 2024)	Perbandingan Random Forest dan SVM dalam Analisis Sentimen Quick Count Pemilu 2024	*Random forest, SVM*	Akurasi *Random Forest* = 78%, Akurasi *SVM* = 80%
(Naufalino, Rianto dan Al-husaini 2025)	Perbandingan Algoritma *Machine Learning* Untuk Analisis Sentimen Secara *Real-Time* Pada Aplikasi *Indodax* Menggunakan *Feature Selection Chi-Square*	*Support Vector Machine*, *Random Forest*, *XGBoost*, *Feature Selection Chi-Square*	Akurasi *Support Vector Machine* = 95%, *Random Forest* = 92%, *XGBoost* = 93%

Pada Tabel 2.3 terdapat beberapa kelemahan dalam penggunaan algoritma *Support Vector Machine*, *Random Forest* dan *XGBoost*, yaitu dalam hasil yang tidak relevan atau redundan (Erkamim dkk. 2023). Solusi untuk mengatasi permasalahan yang terjadi seperti fitur yang tidak relevan atau redundan adalah dengan menambahkan *Feature Selection Chi-Square* seperti penelitian yang dilakukan sebelumnya oleh (Ratiasasadara, Sudarno, dan Tarno 2023). *Chi-Square* adalah pemilihan fitur yang diawasi yang dapat menghilangkan banyak fitur untuk menaikkan tingkat akurasi (Erkamim dkk. 2023). Fitur-fitur yang tidak relevan atau kurang informatif dapat dihilangkan, sehingga meningkatkan efisiensi dan akurasi model dalam analisis sentimen atau klasifikasi lainnya (Ernayanti dkk. 2023).

2.10 Matriks Penelitian

Matriks penelitian menunjukkan perbandingan capaian yang akan didapatkan pada penelitian ini dengan capaian yang telah didapatkan oleh penelitian sebelumnya. Perbandingan ini mengacu pada kinerja model yang menggunakan algoritma *Support Vector Machine*, *Random Forest* dan *XGBoost* dengan penambahan *Feature Selection Chi-Square* dengan algoritma tanpa penambahan *Feature Selection Chi-Square*.

Tabel 2.4 Matriks Penelitian

Penelitian	***Support Vector Machine***	***Random Forest***	***XGBoost***	***Chi-Square***
(Wahyudi dan Kusumawardana 2021)	√	-	-	-
(Jayanti dkk. 2024)	√	√	-	-
(Sondakh dkk. 2023)	√	√	-	-
(Yasir dkk. 2024)	√	-	√	
(Erkamim dkk. 2023b)	-	√	√	-
(Mardiyati dkk. 2023)	-	√	-	-
(Kurniawan dan Indahyanti 2024)	-	-	-	-

(Azizan, Andini, dan Azzahra 2023)	√	√	-	-
(Topaloglu 2024)	√	-	-	√
(D. J. Putri, Dwifebri, dan Adiwijaya 2023)	-	-	√	√
(Tasnim dan Habiba 2021)	√	√	√	√
(Ratiasasadara, Sudarno, dan Tarno 2023)	-	-	-	√
(Septiana dan Alita 2024)	√	√	-	-
(Naufalino, Rianto dan Al-husaini 2025)	√	√	√	√

Tabel 2.4 menunjukkan penelitian terkait yang dijadikan sebagai acuan dalam penelitian ini. Penelitian oleh (Erkamim dkk. 2023) dan (Septiana dan Alita 2024) membahas penggunaan algoritma *Support Vector Machine, Random Forest*, dan *XGBoost* dalam analisis sentimen, namun belum menerapkan *Feature Selection Chi-Square,* sehingga berpotensi menyebabkan redundansi data yang berdampak pada efisiensi dan akurasi model.

Penelitian oleh (Ratiasasadara, Sudarno, dan Tarno 2023) telah menerapkan *Feature Selection Chi-Square* dalam proses analisis sentimen, tetapi terbatas pada

satu algoritma saja, yaitu *Naïve Bayes*, dan hanya menghasilkan akurasi sebesar 80%. Dengan demikian, meskipun sudah ada penelitian yang menggunakan *Chi-Square* atau membandingkan beberapa algoritma *machine learning*, belum ada penelitian yang secara khusus membandingkan performa algoritma *Support Vector Machine*, *Random Forest*, dan *XGBoost* dengan menerapkan *Feature Selection Chi-Square* dalam konteks analisis sentimen aplikasi Indodax.

Penelitian ini akan menghasilkan keterbaruan yaitu membandingkan performa tiga algoritma yaitu *SVM, Random Forest,* dan *XGBoost* secara langsung, dengan penerapan *Feature Selection Chi-Square* untuk meningkatkan efisiensi dan efektivitas klasifikasi sentimen. Selain itu, penelitian ini juga memiliki perbedaan dalam proses akuisisi *dataset*, yaitu pengumpulan *dataset* dilakukan secara *real-time* dari ulasan aplikasi *Indodax* di *Google Play Store*, sehingga menjamin keterkinian dan relevansi data yang digunakan dibandingkan penelitian sebelumnya.