BAB II
TINJAUAN PUSTAKA

2.1 Tinjauan Penelitian Terdahulu

Sebagai rujukan penelitian dalam penyusunan skripsi dengan memaparkan hasil dari penelitian terdahulu berupa jurnal berkaitan dengan tema atau judul yang dijadikan penelitian. Berikut Tabel 2.1 merupakan table dari penelitian terdahulu :

Tabel 2.1 Tinjauan Penelitian Terdahulu

No.	Peneliti	Judul	Tahun	Hasil
1.	Rizqiyani, Mulwinda, Putri	Klasifikasi Judul Buku dengan Algoritma *Naive Bayes dan* Pencarian Buku pada Perpustakaan Jurusan Teknik Elektro.	2017	Mendapatkan hasil akurasi rata-rata sebesar 97,7%.
2.	Akromunnisa, Hidayat	Klasifikasi Dokumen Tugas Akhir (skripsi) Menggunakan *K-Nearest Neighbor*.	2019	Mendapatkan akurasi terbaik sebesar 99,4% pada data 8:2 pada k=1.
3.	Asril, Mustakim, Kamila	Klasifikasi Dokumen Tugas Akhir Berbasis Text Mining menggunakan Metode *Naïve Bayes* Classifier dan *K-Nearest Neighbor.*	2019	*Naïve Bayes* mendapatkan akurasi 86,1%, sedangkan *K-Nearest Neighbor mendapatkan akurasi 91,6%.*
4.	Saputra	Analisis data mining untuk pemetaan mahasiswa yang membutuhkan bimbingan dan konseling menggunakan algoritma *Naïve Bayes*	2018	Mendapatkan akurasi terbesar sebesar 90,5% dengan *Naïve Bayes*.
5.	Nugraha	Analisis Sentimen Berbasis Emoticon pada Komentar Instagram Bahasa Indonesia Menggunakan *Naïve Bayes.*	2021	*Naïve Bayes* mendapatkan akurasi sebesar 96,3%.
6.	Verawati, Audit	Algoritma *Naïve Bayes* Classifier Untuk Analisis	2022	Dapat melakukan analisis sentiment

		Sentiment Pengguna Twitter Terhadap Provider By.u.		dengan algoritma *Naïve Bayes* dan mendapatkan akurasi terbesar sebesar 85%.
7.	Devita, Herwanto, Wibawa	Perbandingan kinerja metode *Naïve Bayes* dan *K-Nearest Neighbor* untuk klasifikasi artikel berbahasa Indonesia.	2018	Pada perbandingan *Naïve Bayes* mendapatkan akurasi sebesar 70% dan *K-Nearest Neighbor* mendapatkan akurasi sebesar 40%.
8.	Manalu, Ginting	Analisis Perancangan Aplikasi Pengklasifikasi Dokumen pada BPKA Deli Serdang Menggunakan Algoritma *Naive Bayes.*	2021	Algoritma *Naïve Bayes* dapat digunakan klasifikasi.
9.	Kosasih, Alberto	Analisis Sentimen Produk Permainan Menggunakan Metode TF-IDF Dan Algoritma *K-Nearest Neighbor.*	2021	Mendapatkan hasil akurasi sebesar 79,3%.
10.	Putri, Ristyawan, Muzaki	Perbandingan Kinerja Algoritma *K-NN* dan *NBC* Untuk Klasifikasi Penyakit Jantung.	2022	*Naïve Bayes* mendapatkan akurasi sebesar 86,1% lebih baik dibanding *K-Nearest Neighbor* yang mendapatkan akurasi sebesar 85,1%

2.2 Teori Dasar yang Digunakan

2.2.1 Klasifikasi

Klasifikasi adalah proses mengelompokkan menurut ciri-ciri tertentu. Sistem Klasifikasi digunakan untuk mengukur objek data yang akan dimasukkan ke dalam kelas-kelas tertentu dari sejumlah kelas yang sudah digunakan atau tersedia(Aji Priyambodo and Prihati Prihati 2020).

Membangun model sebagai prototipe untuk disimpan dalam memori dan menggunakan model ini untuk memperkenalkan, mengklasifikasikan, atau memprediksi objek data baru sehingga dapat ditentukan termasuk kelas mana

yang merupakan dua jenis pekerjaan utama yang dilakukan dalam klasifikasi (Asril, Mustakim and Kamila 2019).

2.2.2 Algoritma *Naïve Bayes*

Algoritma ini merupakan salah satu algoritma yang digunakan pada Teknik klasifikasi. *Naive Bayes* adalah klasifikasi yang dibuat dengan metode probabilitas(Putri, Ristyawan, and Muzaki 2022).

Algoritma ini dahulu dikenal dengan Teorema *Bayes*, karena mengacu pada kemampuan algoritma ini untuk mengantisipasi peluang masa depan berdasarkan pengalaman sebelumnya Menurut klasifikasi *Naive Bayes*, nilai atribut kelas tidak dapat dipengaruhi oleh nilai atribut lainnya. Karena menganggap kemunculan satu kata dalam sebuah kalimat tidak dapat mempengaruhi kemunculan kata lain (Verawati and Audit 2022).

$$p(w_i|c_j) = \frac{1+n_i}{n+|kosakata|}$$

dimana:

$p(w_i|c_j)$: probabilitas kata pada setiap kategori

n_i : frekuensi kemunculan kata setiap kategori

n : jumlah seluruh kata dalam dokumen pada kategori tertentu

$|kosakata|$: jumlah total kata di semua data latih

$$p(c_j) = \frac{n(doc_j)}{n\,(sampel)}$$

dimana:

$p(c_j)$: probabilitas dokumen kategori

$n(doc_j)$: jumlah seluruh dokumen pada suatu kategori

$n\,(sampel)$: jumlah seluruh dokumen latih

2.2.3 Algoritma *K-Nearest Neighbor*

Algoritma ini merupakan sebuah metode untuk melakukan klasifikasi berdasarkan (K) paling dekat dengan objek baru atau paling dekat dapat digunakan untuk mengklasifikasikannya menggunakan algoritma ini (Firdaus, F., Pasnur, P., & Wabdillah 2019).

Karena hasil dari *query instance* atau kategori diurutkan menurut jumlah kelas terbanyak yang muncul, *K-Nearest Neighbor* dimasukkan ke dalam algoritma supervised learning (Fauziah, Sulistyowati, and Asra 2019).

Klasifikasi ketetanggaan digunakan sebagai nilai prediksi dari instance query baru oleh algoritma *K-Nearest Neighbor*. Perhitungan melibatkan

penjumlahan semua nilai kesamaan untuk satu kategori, yang akan dibandinkan dengan nilai mana pun yang lebih tinggi.

$$Cos(i,k) = \frac{\sum_k (d_i d_k)}{\sqrt{\sum_k d_{ik}^2}\sqrt{\sum_k d_{jk}^2}}$$

dimana:

$\sum_k (d_i d_k)$: *vector* dari produk *i* dan *k*
$\sqrt{\sum_k d_{ik}^2}$: panjang dari *vector i*
$\sqrt{\sum_k d_{jk}^2}$: panjang dari *vector j*
i : data uji ke-*i*
j : data latih ke-*j*

2.2.4 *Confusion Matrix*

Dalam mesin learning, *Confusion Matrix* adalah tabel yang sering digunakan untuk mengevaluasi keefektifan model klasifikasi. Jumlah data yang diklasifikasikan dengan benar atau salah dirinci dalam tabel ini(Sani et al. 2022).

Confusion matrix juga merupakan alat analisis prediktif yang dapat digunakan untuk menghasilkan metrik evaluasi seperti akurasi, presisi, daya ingat, dan *F1-Score* atau *F-Measure*. Selain itu, ini menampilkan dan mengontraskan nilai sebenarnya dengan nilai model yang diprediksi (Dwi Pramita B. B1, Ristu Saptono2 2018).

2.2.5 *Python*

Python adalah bahasa pemrograman tingkat tinggi dengan manajemen memori (penunjuk) otomatis, membuatnya mudah dipelajari. Memanfaatkan semantik dinamis dan pemrograman berorientasi objek untuk meningkatkan keterbacaan sintaks, bahasa pemrograman Python tingkat tinggi mampu secara langsung (interpretatif) mengeksekusi sejumlah instruksi multiguna. Karena perintah atau kode program yang digunakan dalam bahasa pemrograman mirip dengan bahasa manusia (bahasa Inggris), maka disebut sebagai bahasa pemrograman tingkat tinggi (*high-level programming language*) (Nugraha 2021).

Python menggunakan pemrosesan yang ditafsirkan, yang menghilangkan kebutuhan untuk proses kompilasi dan memungkinkan kode program diproses baris demi baris langsung dari sumbernya. Ini mirip dengan bahasa *scripting* seperti JavaScript dan PHP (Nugraha 2021).