BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Dalam penelitian ini, penelitian terkait dilakukan untuk mendapatkan pemahaman yang lebih mendalam tentang cara yang digunakan dan untuk memperjelas perbedaan topik penelitian. Berikut dilampirkan pada tabel 2.1 penelitian-penelitian terdahulu yang berkaitan dengan penelitian "Implementasi Algoritma *K-Nearest Neighbor* Berbasis *AdaBoost* Untuk Klasifikasi Penyakit Ginjal Kronis".

Tabel 2. 1 Penelitian Terdahulu

No.	Peneliti	Judul dan Tahun Penelitian	Metode	Masalah	Hasil
1.	Rezi Yuliani [11]	Penerapan Algoritma *C4.5* Berbasis *AdaBoost* Untuk Memprediksi *Financial Distress* Perusahaan, 2020[11]	Algoritma *C4.5* dan *AdaBoost*[11] .	Meminimalisir terjadinya *financial distress*, dilakukan pengamatan dan menganalisa kondisi keuangan dari segi neraca dan laporan laba rugi yang terdapat dalam laporan keuangan[11].	Berdasarkan pengujian yang telah dilakukan dengan rasio data latih dan data uji 90% : 10% hasil *Accuracy* dari Algoritma *C4.5* adalah 72,97% dan setelah ditambah dengan *AdaBoost Accuracy* meningkat menjadi 86,49%.

No.	Peneliti	Judul dan Tahun Penelitian	Metode	Masalah	Hasil
					Algoritma *C4.5* dan *AdaBoost* baik digunakan dalam prediksi *Financial Distress* perusahhan [11].
2.	Komang Anggada Sugiarta[6]	Optimasi *K-Nearest Neighbor* Menggunakan *Bat Algorithm* Untuk Klasifikasi Penyakit Ginjal Kronis, 2019[6].	Algoritma *K-Nearest Neighbor* dan *Bat Algorithm*[6].	Tingginya angka kematian yang terus meningkat pada penderita penyakit ginjal kronis[6].	Dari hasil pengujian dengan mengukur optimasi *K-Nearest Neighbor* menggunakan metode *Bat Algorithm* didapatkan nilai akurasi 100%[6].
3.	Laila Qadrini [20]	*Decision Tree* Dan *AdaBoost* Pada Klasifikasi Penerima Program Bantuan Sosial, 2021[20]	*Decision Tree* dan *AdaBoost*[20]	Tingginya angka kemiskinan yang terjadi di Indonesia, pemerintah mengupayakan penerima program bantuan social[20].	Metode klasifikasi *Decision Tree* dan *AdaBoost* sama-sama memiliki hasil *accuracy* yang baik yaitu 94% dan 95%[20].

No.	Peneliti	Judul dan Tahun Penelitian	Metode	Masalah	Hasil
4.	Hartati[21]	Optimasi Analisis Sentimen Pada *Twitter* Olshop Tokopedia Menggunakan Textmining Dengan Algoritma *Naïve Bayes* dan *AdaBoost, 2022*[21].	*Naïve Bayes* dan *AdaBoost*[21].	Terdapat isu-isu terkait pelayanan maupun opini msyarakat di *twitter* Tokopedia yang bersifat positif dan negative[21].	Algoritma *Naïve Bayes* yang ditambahkan *feature Synthetic Minority Over-sampling Technique* (SMOTE), yang dioptimasi dengan *AdaBoost* mendapatkan nilai *accuracy* 94,95%, *precision* 90,86%, *recall* 100% dan AUC 0,950[21].
5.	Amin Nur Rais[22]	Optimasi Akurasi Klasifikasi Pada Prediksi *Smokte Detection* dengan Menggunakan Algoritma *AdaBoost*, 2022[22].	Algoritma *AdaBoost* dan *Naïve Bayes*[22].	Permasalahan mengenai kebakaran yang menjadi ancaman bagi alam dan lingkungan[22].	Hasil penilitian menunjukkan algoritma AdaBoost meningkatkan nilai akurasi 98,70%, presisi 97,10%[22].

2.2 Landasan Teori

Penelitian terkait klasifikasi penyakit ginjal kronis, yang menerapkan metode *K-Nearest Neighbor* berbasis *AdaBoost* memerlukan berbagai bidang keilmuan untuk membantu penelitian ini, berikut landasan teori yang dibutuhkan.

2.2.1 Penyakit ginjal kronis

Persoalan penyakit ginjal kronis kerap kali mengalami peningkatan yang terjadi pada keadaan jasmani manusia secara global[1]. Penyakit ginjal kronis hadir lebih dari tiga bulan sebagai abnormalitas struktural ginjal dengan atau tanpa penurunan laju filtrasi glomerulus (GFR), didapati kelainan patologis dan ditandai kerusakan ginjal yang dapat berupa anomali laboratorium darah atau urin maupun anomali radiologis[23]. Dengan LPG <60 mL/menit/1,73 m^2 selama kurang dari tiga bulan dan dapat disertai atau tanpa disertai kerusakan ginjal[23]. Ditemukan sebagian faktor risiko penyebab penyakit ginjal kronis seperti *hipertensi*[23], *diabetes melitus*[23], riwayat keluarga penyakit ginjal kronis[23] dan penyakit *kardiovaskular*[23]. Berikut data [19] yang digunakan dalam klasifikasi penyakit ginjal kronis dengan adanya pembagian kategori berdasarkan atribut untuk perhitungan *K-Nearest Neighbor* berbasis *AdaBoost*.

Tabel 2. 2 Kategori Berdasarkan Atribut Untuk Pergitungan Manual *K-Nearest Neighbor* berbasis *AdaBoost*

No.	Atribut	Keterangan Pembagian Kategori
1.	*Age*[19]	Usia 2 tahun ≤ 25 tahun, 26 tahun ≤ 49 tahun, 50 tahun ≤ 82 tahun[19], [26].
2.	*Blood Pressure* (BP)[19]	Tekanan darah rendah ≤ 90 mmHg, normal ≤ 120 mmHg hipertensi stadium 2 ≥ 140 mmHg, krisis hipertensi ≥ 180 mmHg[19], [27], [28], [29], [30].

3.	*Specific Gravity* (SG)[19]	*Specific gravity* ≤ 1.010 yaitu mengencerkan urine, > 1.010 yaitu memekatkan urine[19], [31].
4.	*Albumin* (AL)[19]	Kadar *albumin* normal 3 ≤ 5 g/dL, kadar *albumin* rendah < 3 g/dL[19], [32].
5.	*Sugar* (SU)[19]	*Sugar* normal 4 ≤ 5 mmol/L, rendah ≤ 3 mmol/L[19], [33].
6.	*Red Blood Cells* (RBC)[19]	*Normal, abnormal*[19].
7.	*Pus Cell* (PC)[19]	*Normal, abnormal*[19].
8.	*Puss Cell Clumps* (PCC)[19]	*Present, notpresent*[19].
9.	*Bacteria* (BA)[19]	*Present, notpresent*[19].
10.	*Blood Glucose Random* (BGR)[19]	*Blood Glucose Random* rendah < 70 *mg/dl*, normal < 200 *mg/dl*, ≥ 200 *mg/dl*[19], [33].
11.	*Blood Urea* (BU)[19]	*Blood Urea* normal 20 ≤ 40 *mg/dl*, rendah < 20 *mg/dl*, > 40 *mg/dl*[19], [34].
12.	*Serum Creatinine* (SC)[19]	*Serum Creatinine* normal 0,6 ≤ 1,3 *mg/dl*, rendah < 0,6 *mg/dl*, tinggi > 1,3 *mg/dl*[19], [35].
13.	*Sodium* (SOD)[19]	*Hyponatremia* < 135 *mmol/L*, normal 135 ≤ 145 *mmol/L*, *hypertenatremia* > 145 *mmol/L*[19], [36].
14.	*Potassium* (POT)[19]	*Hypokalemia* < 3,5 *mEq/L*, normal 3,5 ≤ 5,0 *mEq/L*, *hyperkalemia* 5,1 ≤ 6,0 *mEq/L*, *dangerously high* > 6,0 *mEq/L*[19], [37].

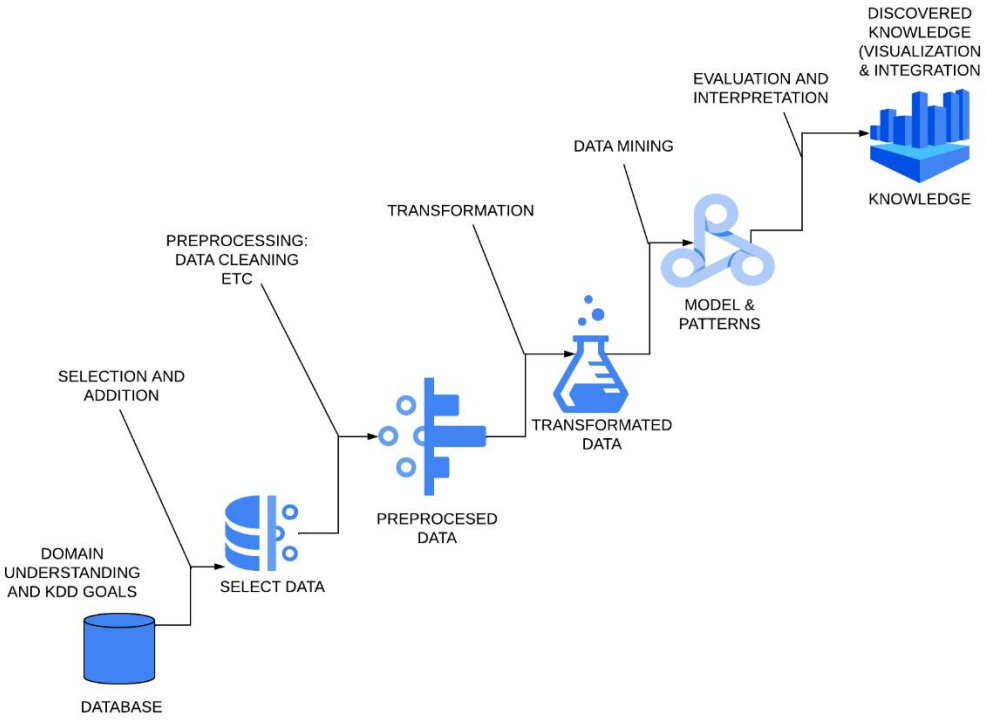
15.	*Hemoglobin* (HEMO)[19]	*Hemoglobin* 12 ≤ 18 *g/dl, anemia* < 12 *g/dl, polisitemia* > 18 *g/dl*[19], [38].
16.	*Packed Cell Volume* (PCV)[19]	*Packed Cell Volume* rendah < 35, normal 35 ≤ 47, tinggi > 47[19], [39].
17.	*White Blood Cell Count* (WC)[19]	*Leukopenia* < 4000, normal 4000 ≤ 11000, *leukositosis* > 11000[19], [40].
18.	*Red Blood Cell Count* (RC)[19]	*Mikrositik* < 4,5, normal 4,5 ≤ 6, *makrositik* > 6[19], [41].
19.	*Hypertension* (HTN)[19]	*Yes, no*[19]
20.	*Diabetes Mellitus* (DM)[19]	*Yes, no*[19]
21.	*Coronary Artery Disease* (CAD)[19]	*Yes, no*[19]
22.	*Appetite* (APPET)[19]	*Good, poor*[19].
23.	*Pedal Edema* (PE)[19]	*Good, poor*[19]
24.	*Anemia* (ANE)[19]	*Yes, no*[19]

2.2.2 *Data mining*

Tahap perencanaan *data mining* memuat informasi yang akurat untuk perkiraan berdasarkan tren masa lalu dan kondisi saat ini[42]. *Data mining* merupakan analisis melihat gabungan data untuk mendeteksi kesinambungan yang tidak terduga dan meringkas data dengan cara yang berbeda dari sebelumnya untuk mudah dimengerti dan berguna bagi pemilik data[43]. Dalam beberapa kasus, banyak teknik yang termasuk dalam *data mining* dapat dieksploitasi, salah satunya adalah teknik klasifikasi, klasifikasi itu sendiri adalah bentuk dasar dari analisis data[44].

Berdasarkan kategori kegunaan *data mining* meliputi deskriptif dan prediktif, deskriptif didefinisikan sebagai menemukan pola dan kesinambungan yang tidak

diketahui dalam kumpulan data, dan *data mining* yang dimanfaatkan untuk membentuk sebuah model pengetahuan dalam klasifikasi adalah prediktif, beberapa operasi *data mining* yang termasuk deskriptif yaitu *clustering*, *summarization*, dan *sequence discover*, pada kegunaan *data mining* yang termasuk prediktif adalah klasifikasi, regresi, *time series analysis*, dan *prediction*[45]. *Data mining* memiliki beberapa cara yang dimanfaatkan pada alur aplikasi penambangan data, salah satu strategi yang umum dimanfaatkan adalah *Knowledge Discovery Databases* (KDD) yang terdiri dari 9 fase yang ditampilkan pada Gambar 2.1[46].



Gambar 2. 1 Tahapan *Knowledge Discovery Databases* (KDD)[46]

Setiap langkah *Knowledge Discovery Databases* (KDD), dijelaskan sebagai berikut[46]:

1. *Domain understanding and KDD Goals*

   Sasaran ditentukan dari sudut pandang pengguna dalam membantu mengembangkan dan memahami aplikasi dan informasi sebelumnya[46].

2. *Selection and additions*

Langkah kedua, berpusat mendefinisikan data target dan himpunan bagian dari sampel atau data atribut, tahap pembagian data latih dan uji yang akan diolah menggunakan rasio 70:30[47].

3. *Preprocessing*

Prapemrosesan data adalah operasi mendasar untuk menyelesaikan data yang konsisten tanpa gangguan *missing value*[46]. Menangani *missing value* dengan metode imputasi[48]. Mengganti nilai kosong dengan nilai rata-rata, dari nilai atribut berdasarkan masing-masing jenis atribut[49]. Kedua, menangani dengan menghapus data atribut, dengan syarat data kosong lebih dari 20%, karena semakin tinggi data kosong yang diisi mempengarusi keakuratan data, menggunakan rumus persentase untuk mengetahui berapa persen missing value yang ada pada atribut, dihitung menggunakan rumus berikut[50].

$$P = \frac{f}{n} \times 100\% \quad (2.1)$$

Keterangan:

P = persentase

f = frekuensi data *missing value*

n = jumlah keseluruhan data

100% = angka tetap

Rumus untuk mengetahu nilai rata-rata, yaitu[51]:

$$\bar{X} = \frac{\sum X_i}{n} \quad (2.2)$$

Keterangan :

$\bar{X}$ = nilai rata – rata

X = nilai data

n = banyak data

4. *Transformation*

Mengkonversi nilai nominal ke tipe data numerik, sehingga informasi mudah diimplementasikan[46].

5. *Data mining* (*chosing the suitable data mining task*)

Pilih algoritma penambangan data yang sesuai berdasarkan tujuan spesifik yang ditentukan di fase awal, contohnya teknik *data mining* adalah klasifikasi, regresi, pengelompokan dan peringkasan[46].

6. *Data mining* (*chosing the suitable data mining algorithm*)

Saat menentukan algoritma yang tepat untuk menemukan pola data, algoritma yang dipilih didasarkan pada konsistensi metode *data mining*[46].

7. *Data mining* (*imploying data mining algorithm*)

Mengaplikasikan algoritma yang telah dipilih[46].

8. *Evaluation and interpretation*

Fase ini berfokus pada interpretasi dan evaluasi, memeriksa pola atau informasi yang ditemukan yang mungkin atau mungkin tidak bertentangan dengan hipotesis yang sudah ada sebelumnya[46].

9. *Discovered knowledge*

Dalam proses KDD, informasi yang ditemukan dapat dimanfaatkan untuk memutuskan apa yang harus dilakukan dengan informasi yang dihasilkan[46].

2.2.3 Klasifikasi

Klasifikasi adalah bagian metode pembelajaran terawasi, yang membutuhkan data pembelajaran berlabel pembelajar untuk memperkirakan kelas dari objek yang tidak diketahui[52]. Klasifikasi memiliki nilai data akurasi, atau kesamaan hasil pengukuran dengan bilangan real atau data[53], hasil dari jumlah data uji yang benar dan data uji yang salah pada algoritma adalah nilai dari *accuracy*, *recall* dan *precision* yang diperoleh menggunakan *confusion matrix*[54]. Akurasi yang tinggi

diartikan sebagai klasifikasi yang berhasil dalam menyelesaikan suatu kasus dan dapat diklasifikasikan sebagai klasifikasi yang akurat dan efisien[52].

2.2.4 Algoritma *K-Nearest Neighbor*

Algoritma *K-Nearest Neighbor* (K-NN) termasuk algoritma *supervised learning*[55]. Sistem kinerja dari *supervised learning* adalah dari data pelatihan yang diberikan, sistem akan mempelajari data yang telah ada, sistem akan mempelajari pola dari data sebagai acuan untuk data selanjutnya[56]. Algoritma *K-Nearest Neighbor* adalah unsur pengklasifikasian objek berdasarkan data latih yang paling dekat dengan objek tersebut, data latih tersebut dijelaskan dengan atribut numerik n-dimensi, pada setiap data latih memiliki titik di n-dimensi mewakili ruang, diberikan data kueri yang pengidentifikasinya tidak diketahui, dengan *K-Nearest Neighbor* mencari k data pelatihan yang paling dekat dengan data kueri dalam ruang n-dimensi, jarak antara data kueri dan data pelatihan dihitung dengan mengukur jarak antara titik yang mewakili data kueri dan semua titik yang mewakili data pelatihan[57]. Jumlah tetangga yang digunakan sama dengan k tetangga, ambil k tetangga terdekat terlebih dahulu, hitung data terdekat dari k tetangga terdekat dari kelas yang ada, kelas dengan data terbaru menjadi kelas pemenang yang ditetapkan sebagai kelas pengenal dalam data X[58].

Data numerik mampu ditangani oleh algoritma *K-Nearest Neighbor*, namun tipe non-numerik akan dikonversi atau dimasukkan ke dalam *preprocessing* data[59]. Sebelum melakukan perhitungan, langkah awal yang diambil yaitu dengan menentukan jumlah tetangga (k) dalam pertimbangan penentuan kelas, untuk menentukan nilai k tidak memiliki rumus pasti, menentukan nilai k dengan mempertimbangkan jumlah data yang ada, jumlah data yang genap menggunakan nilai k ganjil dan jumlah data yang ganjil menggunakan nilai k genap, dari *ranking* data yang dilakukan mentukan kelas data baru dari hasil kelas mayoritas yang diperoleh[60]. Tahapan dari algoritma K-Nearest Neighbor, pertama siapkan data latih, tahap kedua mentukan nila k, ketiga menghitung jarak terdekat dapat menggunakan jarak *euclidean,* lalu meranking hasil perhitungan *euclidean,* tahap

selanjutnya menghitung jumlah titik pada k untuk setiap kelas dan tahap akhir mengklasifikasikan data baru ke kelas k terbanyak[61].

Metode jarak *euclidean* merupakan perhitungan jarak pendekatan dalam mengukur dua titik yang berbeda[62]. Metode pendekatan *K-Nearest Neighbor* yang digunakan pada penelitian ini, yaitu metode pendekatan jarak *euclidean*[63]. Persamaan perhitungan untuk mencari jarak *euclidean* adalah sebagai berikut[64]:

$$d_i = \sqrt{\sum_{1=1}^{p}(X_{1i} - X_{2i})^2} \quad (2.3)$$

Keterangan:

X_1: sample data uji

X_2: data uji

d : jarak antara X dan Y

p : dimensi data

i : setiap data

2.2.5 *AdaBoost*

Adaptive Boost (*AdaBoost*) adalah unsur varian dari algoritma *boosting*, *boosting* (*AdaBoost*) didefinisikan sebagai pendekatan pembelajaran mesin untuk meningkatkan aturan perkiraan yang akurat dengan menggabungkan aturan yang relatif lemah dan tidak tepat[65]. Penerapan algoritma *AdaBoost* dapat dikombinasikan dengan algoritma klasifikasi lainnya untuk memperoleh hasil kinerja klasifikasi yang baik, yaitu dengan kinerja bobot dilampirkan pada setiap pelatihan dan kemudian pengklasifikasi yang berbeda dilatih menggunakan bobot tersebut[11]. Di bawah ini adalah tahapan dari algoritma *AdaBoost*[20]:

1) Input: Suatu kumpulan sample penelitian dengan label $\{(x_i, y_i), \dots, (x_N, y_N)\}$ suatu *component learn* algoritma, jumlah perputaran T.
2) *Initialize*: Bobot suatu sampel pelatihan $w_i^1 = {}^1/_N$, untuk semua i = 1, … , N
3) *Do for* t = 1, … , T

4) Gunakan *component learn* algoritma untuk melatih suatu komponen klasifikasi, pada sample bobot pelatihan.
5) Hitung kesalahan pelatihannya pada h_t:$\varepsilon_t = \sum_i^N w_i^t, y_i \neq h_i(x_i)$
6) Tetapkan bobot untuk *component classifier* $h_t = \alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$
7) *Update* bobot sample pelatihan $w_i^{t+1} = \frac{w_i^t \exp\{-\alpha_{th_t(x_i)}\}}{C_t}$, i = 1,…,*N* C_t adalah suatu konstanta normalisasi.
8) *Output*: *f* (X) = sign $(w_{t=1}^T \alpha_t h_t(x))$

2.2.6 *Label encoding*

Label-encoding bekerja dengan mengubah data menjadi bentuk numerik, yang diubah menjadi bentuk yang dapat dibaca komputer atau diproses dengan memberikan setiap kategori data nomor unik, yang dimulai dari 0[16]. Dan fungsi *One-hot encoding* mengklasifikasikan pengenal kelas yang ada menjadi bilangan biner, angka 1 berarti pengenal kelas ada dan angka 0 berarti pengenal kelas tidak ada[66]. *Label encoding* merupakan bagian dari *pre-processing*[67].

2.2.7 Entropi

Entropi digunakan untuk menentukan nilai perolehan informasi, entropi menjelaskan jumlah informasi yang dibutuhkan untuk mengkodekan suatu kelas [68]. Semakin rendah nilai entropi, semakin baik digunakan untuk mengekstrak kelas[69]. Nilai entropi dapat dihitung dengan menggunakan persamaan berikut [70]:

$$Entropi\ S = \sum_{i=1}^{n} - pi \times log_2 pi \qquad (2.4)$$

Keterangan:

S : Himpunan kasus

n : Jumlah partisi S

pi: Proporsi dari Si terhadap S

2.2.8 *Information gain*

Information gain (IG) didefinisikan sebagai teknik pemilihan fitur yang menggunakan metode pembobotan atribut nominal atau kontinyu yang didiskritisasi menggunakan entropi maksimum, istilah perolehan informasi diukur dengan menghitung jumlah bit informasi dari kelas yang diklasifikasi, dengan dokumen tersebut mengandung term atau tidak, teknik pemilihan fitur *information gain* berarti pemilihan simpul fitur dari pohon keputusan berdasarkan nilai *information gain*, nilai *information gain* suatu fitur ditentukan oleh efek dari fitur tersebut, pada satuan kelas yang diukur dalam data dibagi menjadi sub-data dengan nilai karakteristik tertentu[68]. Pemilihan atribut pada *root* didasarkan pada nilai validasi tertinggi dari atribut yang ada, *gain* (S.A) adalah kumpulan informasi tentang atribut A dalam kaitannya dengan informasi awal S, berikut adalah persamaan untuk menghitung nilai gain[70]:

$$Gain\ S, A = \sum_{i=1}^{n} \frac{Si}{s} \times Entropy\ Si \tag{2.5}$$

Keterangan:

S : Himpunan kasus

A : Atribut

n : Jumlah partisi atribut A

|Si|: Jumlah kasus pada partisi ke-i

|S| : Jumlah kasus dalam S

2.2.9 *Split Data*

Proses klasifikasi membutuhkan data yang telah dibagi terlebih dahulu atau disebut juga *split data*, terdiri dari dua jenis, yaitu data latih (*data training*) dan data uji (*data testing*)[71]. Pembagian data memiliki rasio perbandingan, pada penelitian ini akan menggunakan 70% data latih dan 30% data uji[72]. Menggunakan jumlah data latih lebih banyak, untuk mempermudah pengujian kebenaran dan keakuratan pada data uji[73].

2.2.10 *Confusion matrix*

Teknik yang digunakan untuk mencari nilai akurasi dengan menggunakan tabel yang menjelaskan jumlah data uji yang benar dan salah disebut *confusion matrix* [74]. *Confusion matrix* menjadi alat visualisasi yang umum digunakan dalam pembelajaran terawasi, setiap kolom matriks adalah contoh kelas yang diklasifikasi dan setiap baris mewakili kejadian kelas yang sebenarnya[75]. Setiap sel berisi angka menunjukkan berapa banyak kasus yang sebenarnya dari kelas yang diamati untuk diperkirakan, TP sebagai jumlah *record positive* yang diklasifikasikan sebagai positif, FP sebagai jumah *record negative* yang diklasifikasikan sebagai positif, TN sebagai jumlah *record positive* yang diklasifikasikan sebagai negatif, dan FN sebagai jumlah *record negative* yang diklasifikasikan sebagai negatif[76].

PREDICTED CLASS

ACTUAL CLASS	YES	NO	TOTAL
YES	TP	FN	P
NO	FP	TN	N
TOTAL	P'	N'	P + N

Gambar II.1 *Confusion Matrix*[76]

2.2.11 *Accuracy, recall, precision*

Ada tiga nilai yang dimanfaatkan untuk mengukur kemampuan suatu sistem klasifikasi yaitu *precision*, *recall* dan *accuracy*[77]. Berikut adalah prosedur *confusion matrix* dengan beberapa parameter untuk mengevaluasi kinerja model klasifikasi yang dihasilkan[78][79][80]:

1. *Precision*

   Precision adalah tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh system menggunakan persamaan (2.8), pada persamaan (2.6) yaitu *precision* (*yes*) merupakan

nilai evaluasi dari *precision* yang menghitung kelas positif dan pada persamaan (2.7) yaitu *precision* (*no*) merupakan nilai evaluasi yang menghitung kelas negatif. Berikut rumus persamaan (2.6), (2.7) dan (2.8) ada di bawah ini:

$$Precision\ (yes) = \frac{TP}{TP+FP} \tag{2.6}$$

$$Precision\ (no) = \frac{TN}{TN+FN} \tag{2.7}$$

$$Precision = \frac{Precision(yes)+Precision(no)}{2} \tag{2.8}$$

2. *Recall*

   Recall merupakan tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi dihitung menggunakan rumus persamaan (2.11), pada persamaan (2.9) merupakan sensivisitas yang merupakan nama lain dari *recall* untuk menghitung kelas positif dan pada persamaan (2.10) merupakan spesifisitas yang merupakan nama lain dari *recall* untuk menghitung kelas negatif. Berikut persamaan (2.9), (2.10) dan (2.11) ada dibawah ini:

$$Sensivisitas = \frac{TP}{TP+FP} \tag{2.9}$$

$$Spesifisitas = \frac{TN}{TN+FP} \tag{2.10}$$

$$Recall = \frac{Sensivisitas+Spesifisitas}{2} \tag{2.11}$$

3. *Accuracy* didefinisikan sebagai tingkat kedekatan antara nilai klasifikasi dengan nilai aktual dengan menggunakan rumus persamaan di bawah ini:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{2.12}$$

2.2.12 *Rapidminer*

Rapidminer adalah perangkat lunak yang dikembangkan oleh Dr. Markus Hofmann dari *Institute of Technology Blanchardstown* dan Raif Klinkenberg dari *rapid-i.com* yang menggunakan GUI (*Graphical User Interface*) yang membuat pengguna tidak sukar dalam menggunakan *software* tersebut[81]. *Rapidminer* menjadi bagian dari salah satu *software* dalam pengolahan *data mining*[82]. Pengolahan pada tahap awal yaitu dengan merakit dan memodelkan operator dalam suatu proses sesuai dengan teknik penambangan data yang disediakan oleh alat *Rapidminer*[83].