BAB 2

LANDASAN TEORI

2.1. Penelitian Terkait

Tabel 2. 1 Penelitian Terkait

Penelitian 1	
Penulis	(Rachman & Handayani, 2021)
Judul	Klasifikasi Algoritma *Naive Bayes* Dalam Memprediksi Tingkat Kelancaran Pembayaran Sewa Teras UMKM
Metode	Algoritma *Naive Bayes*
Hasil Penelitian	Hasil pengolahan data pada aplikasi *Rapidminer* dengan penambahan algoritma *Naive Bayes* menghasilkan akurasi 81.81%, *Percision* 66.66%, *Recall* 100% dan nilai *under the curva* (AUC) 0.800.
Penelitian 2	
Penulis	(Guntur et al., 2022)
Judul	Perbandingan *Particle Swarm Optimization* dengan *Genetic Algorithm* dalam *Feature Selection* untuk Analisis Sentimen pada Permendikbudristek PPKS-LPT
Metode Penelitian	*Particle Swarm Optimization, Genetic Algorithm*
Hasil Penelitian	Hasil dari penggunaan *Particle Swarm Optimization* mampu meningkatkan akurasi sebesar 0,8%, Sedangkan penggunaan *Genetic Algorithm* masih belum menemukan kombinasi fitur yang lebih baik.
Penelitian 3	
Penulis	(Amrin et al., 2021)
Judul	Optimasi Algoritma C4.5 dan *Naive Bayes* Berbasis *Particle Swarm Optimization* Untuk Diagnosa Penyakit Peradangan Hati

Metode	Algoritma C4.5, Algoritma *Naive Bayes, Particle Swarm Optimization*
Hasil Penelitian	Pengunaan C4.5 menghasilkan akurasi 70,99% dan nilai *under the curva* (AUC) sebesar 0,950. Sedangkan pengunaan *Naive Bayes* menghasilkan akurasi 66,14% dengan nilai *under the curva* (AUC) sebesar 0,742. Setelah di optimasi, Pengunaan algoritma C4.5 dengan *Particle Swarm Optimization* (PSO) merupakan metode terbaik dengan akurasi 79,51% dan nilai *under the curva* (AUC) sebesar 0,950. Sedangkan *Naive Bayes* dengan *Particle Swarm Optimization* (PSO) menghasilkan akurasi 79,28% dan nilai *under the curva* (AUC) sebesar 0,739. Pembuktian bahwa *Particle Swarm Optimization* (PSO) dapat meningkatkan nilai akurasi dari metode yang digunakan.
Penelitian 4	
Penulis	(Sugianto et al., 2018)
Judul	Klasifikasi Keminatan Menggunakan *Algoritme Extreme Learning Machine* dan *Particle Swarm Optimization* untuk Seleksi Fitur
Metode Penelitian	*Algoritme Extreme Learning Machine, Particle Swarm Optimization (PSO)*
Hasil Penelitian	Hasil akurasi pada klasifikasi *algoritme Extreme Learning Machine* dan *Particle Swarm Optimization* (PSO) menghasilkan akurasi sebesar 94.44%. Sedangkan hasil akurasi jika hanya menggunakan metode *Extreme Learning Machine* tanpa seleksi fitur *Particle Swarm Optimization* (PSO) hanya mencapai 66.67%. Hal ini menunjukkan bahwa metode seleksi

	fitur *Particle Swarm Optimization* (PSO) mampu meningkatkan akurasi untuk permasalahan klasifikasi keminatan.
Penelitian 5	
Penulis	(Arifin, 2020)
Judul	Optimasi *Decision Tree* Menggunakan *Particle Swarm Optimization* Untuk Klasifikasi Sel *Pap Smear*
Metode Penelitian	*Decision Tree, Particle Swarm Optimization*
Hasil Penelitian	Penggunaan algoritma *Decision tree* menghasilkan akurasi sebesar 91.39 % dan nilai *under the curva* (AUC) 0.858, sedangkan pada penerapan algoritma *Particle Swarm Optimization* pada *Decision tree* menghasilkan akurasi yang lebih baik yaitu sebesar 96.76 % dan nilai *under the curva* (AUC) 0.926.

Berdasarkan uraian diatas, Algoritma telah banyak digunakan dalam berbagai studi kasus. Namun, yang membedakan pada penelitian ini dengan penelitian sebelumnya yaitu dimana pada penelitian ini menggunakan atribut dari data banjir Kota Samarinda periode tahun 2019 hingga periode tahun 2022 yang mencakup (temperatur minimum, temperatur maksimum, temperatur rata-rata, kelembapan rata-rata, curah hujan, lamanya penyinaran matahari, kecepatan angin maksimum, arah angin saat kecepatan maksimum, kecepatan angin rata-rata, arah angin terbanyak) dan akan dianalisis dengan dua metode yaitu metode *Naive Bayes* dan *Naive Bayes* dengan penambahan optimasi metode *Particle Swarm Optimization (PSO)* untuk meningkatkan hasil akurasi. Oleh karena itu, mengingat topik penelitian dan lokasi penelitian yang berbeda, maka peneliti tertarik untuk melakukan penelitian dengan menerapkan *Particle Swarm Optimization (PSO)* untuk meningkatkan akurasi algoritma *Naive Bayes* pada banjir di Kota Samarinda.

2.2. Banjir

Banjir merupakan fenomena alam yang disebabkan oleh intensitas hujan yang tinggi atau lebihnya debit air yang tidak dapat ditampung berdasarkan kapasitasnya (santoso dian, 2019). Secara sederhana, banjir dapat didefinisikan sebagai adanya air di suatu wilayah yang luas sehingga menutupi permukaan bumi di wilayah tersebut. Terdapat beberapa jenis banjir diantaranya yaitu : (Putra et al., 2019).

1. Banjir air

   Disebabkan oleh luapan air dari sungai, danau atau selokan sehingga menyebabkan air meluap dan menggenangi daratan. Terjadinya banjir ini dapat disebabkan oleh hujan terus menerus dan menyebabkan sungai atau danau tidak lagi dapat menampung air.

2. Banjir bandang

   Banjir bandang terjadi di tempat-tempat yang dekat dengan pegunungan, dimana tanah pegunungan tampak seperti longsoran yang disebabkan oleh air hujan, yang kemudian dibawa oleh air ke dataran yang lebih rendah. Biasanya banjir ini menghanyutkan beberapa pohon hutan atau bebatuan besar. Dengan itu, tentunya dapat merusak pemukiman penduduk di kawasan pegunungan tersebut.

3. Banjir rob (laut pasang)

   Banjir rob disebabkan oleh naiknya air laut. Air yang naik ini biasanya menahan air sungai yang terkumpul dan nantinya dapat merusak tanggul dan menggenangi daratan.

2.3. Data Mining

Data *Mining* adalah sebuah proses yang menggunakan teknik statistik, *machine learning*, kecerdasan buatan, matematika untuk mengekstraksi dan mengidentifikasi informasi yang berguna dan pengetahuan yang relevan dari database besar (Utomo & Mesran, 2020). Data *Mining*, sering disebut *knowledge discovery in database (KDD)* merupakan kegiatan yang diantaranya pengumpulan, menggunakan data historis untuk menemukan keteraturan, pola atau hubungan

dalam data dalam jumlah besar. Hasil dari data *mining* dapat digunakan untuk memperbaiki pengambilan keputusan di masa mendatang, sehingga istilah *pattern recognition* sudah jarang digunakan karena merupakan bagian dari data *mining* (Sibarani, 2020).

Ada beberapa proses data *mining* yang harus dilakukan sebelum menerima informasi baru yaitu pembersihan data, pembersihan proses menghilangkan *noise* dan data yang tidak konsisten, proses integrasi data yang menggabungkan data dari suatu sumber data yang berbeda, pemilihan data dan proses seleksi (Mandar, 2022). Data *mining* dibagi menjadi beberapa kelompok yaitu : (Naldy & Andri, 2021).

1. *Description* (Deskripsi)

   Sebuah deskripsi dari pola dan kecendrungan sering memberikan penjelasan untuk sebuah pola atau kecendrungan dalam data. Misalnya pada petugas pengumpulan suara tidak ditemukan pernyataan atau fakta siapa yang kurang professional akan sedikit di dukung dalam pemilihan presiden.

2. *Clustering* (Pengklusteran)

   Clustering merupakan pengelompokan *record,* kumpulan data yang menarik pengamatan atau perhatian dan membentuk kelas objek yang serupa. *Cluster* adalah kumpulan *record* yang serupa dengan lainnya dan memiliki perbedaan dengan *record-record* pada *cluster* lain. *Clustering* berbeda dengan klasifikasi yaitu tidak memiliki variabel target dalam pengklusteran.

3. *Association* (Asosiasi)

   Association atau asosiasi digunakan untuk mengidentifikasi kejadian atau proses tertentu, dengan atribut yang muncul di setiap kejadian. Teknik ini mencoba menyusun nilai yang muncul secara bersamaan pada setiap baris dan menampilkan hasil keluaran dalam sebuah *rule*. *Association* atau asosiasi memiliki 2 parameter :

   1) *Support* adalah *persentase* kombinasi atribut dalam basis data.
   2) *Confidence* adalah kekuatan hubungan antara atribut dalam aturan asosiasi

4. *Classification* (Klasifikasi)

   Klasifikasi adalah proses menemukan model atau fungsi yang mendeskripsikan data dan membedakannya ke dalam kelas-kelas. Klasifikasi melibatkan pemeriksaan karakteristik suatu objek dan mengklasifikasikan objek ke dalam salah satu kategori yang telah ditentukan sebelumnya. Klasifikasi terdiri target variabel kategori, contohnya pada klasifikasi pendapatan dapat dibagi menjadi tiga kategori yaitu pendapatan tinggi, dan pendapatan menengah dan pendapatan rendah.

5. *Estimation* (Estimasi)

   Estimasi hampir sama dengan klasifikasi hanya saja pada variabel target etimasi lebih mengarah ke angka bukan kategori. Pemodelan dilakukan menggunakan *record* lengkap dimana nilai dari variabel target menjadi nilai prediksi. Selanjutnya dilakukan penginjauan estimasi nilai dari variable prediksi.

6. *Prediction* (Prediksi)

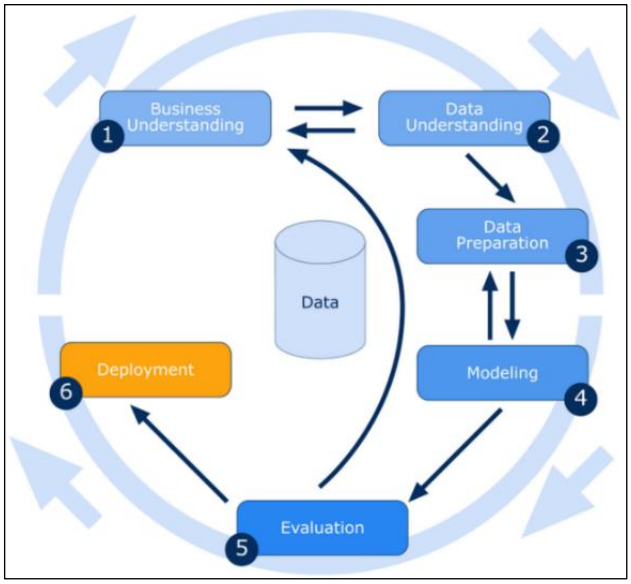
   Prediksi juga hampir sama dengan klasifikasi, tetapi data akan diklasifikasikan berdasarkan perilaku atau nilai yang hasilnya di masa yang akan datang. Misalnya perkiraan perekonomian harga beras tiga bulan mendatang, prediksi tingkat pengangguran 5 tahun yang akan datang.

2.4. CRISP-DM

Cross-Industry Standard Process for Data Mining(CRISP-DM) merupakan sebuah model atau *framework* dalam data *mining* yang awalnya (1996) dibangun oleh 5 perusahaan yaitu *Integral Solutions Ltd* (ISL), *Teradata*, *Daimler AG*, *NCR Corporation dan OHRA. Framework* ini kemudian dikembangkan dengan ratusan organisasi dan perusahaan di Eropa sebagai *methodology standard nonproprietary* bagi data *mining* (Wiratama & Pradnya, 2022).

Cross-Industry Standard Process for Data *Mining* atau CRISP-DM adalah standarisasi proses data *mining* menjadi taktik pemecahan perkara secara generik menurut usaha atau unit penelitian (Feblian & Daihani, 2017). CRISP-DM memiliki siklus hidup yang terdiri dari enam tahap dan tahap-tahapan tersebut bersifat

adaptif, yaitu tahap selanjutnya bergantung dengan hasil yang terkait pada tahap sebelumnya. Hasil yang bergantungan paling signifikan antar tahapan ditunjukkan pada panah digambar. Berikut gambaran siklus pada CRISP-DM : (Hasanah et al., 2021).



Gambar 2. 1 Tahapan CRISP-DM
Sumber : (Hasanah et al., 2021)

1. *Business Understanding* (Pemahaman Bisnis)

   Beberapa hal yang akan dilakukan pada tahap ini, yaitu memahami kebutuhan dan tujuan dari perspektif bisnis serta mengartikan pengetahuan ke dalam bentuk definisi masalah dalam data *mining*, lalu mengidentifikasi rencana dan strategi untuk mencapai tujuan data *mining*.

2. Data *Understanding* (Pemahaman Data)

   Tahapan ini diawali dengan mengumpulkan data, penggunaan analisis eksplorasi data, melakukan evaluasi kualitas data.

3. Data *Preparation* (Pengolahan Data)

   Pada tahap ini, dataset final dibangun dari data mentah. Ada beberapa hal yang harus dilakukan termasuk pembersihan data (Data *Cleaning*), melakukan seleksi data (Data *Selection*), *record* dan atribut-atribut, dan juga melakukan transformasi data (Data *Transformation*) yang digunakan sebagai input pada tahap pemodelan.

4. *Modeling* (Pemodelan)

   Pada tahap ini langsung melibatkan *machine learning* dalam menentukan teknik data *mining*, alat bantu data *mining* serta algoritma data *mining*.

5. *Evaluation* (Evaluasi)

   Tahapan ini dilakukan dengan melihat tingkat kinerja pola yang dihasilkan algoritma untuk menghasilkan kualitas dan efektivitas sebelum publikasi, menentukan apakah model telah memenuhi tujuan atau tidak, menentukan apakah masalah atau penelitian yang belum tertangani dengan baik, membuat keputusan yang relevan menggunakan hasil yang diperoleh dari data *mining*.

6. *Deployment* (Penyebaran)

   Tahapan ini dilakukan dengan membuat laporan menggunakan model yang telah dihasilkan.

2.5. *Naive Bayes*

Naive Bayes Classifier (NBC) merupakan algoritma yang sangat sederhana dan telah banyak digunakan pada teknik data *mining* dalam menerapkan teori *Bayes* dan klasifikasi (Tasya, 2021). Klasifikasi *Naive Bayes* adalah pengklasifikasi pada data *mining* berdasarkan teorema *Bayes*, dengan metode statistik independen dan probabilitas (Monowati & Setyadi, 2023). Pengklasifikasi *Naive Bayes* juga menghasilkan nilai akurasi dan kecepatan tinggi pada *database* besar. Menerapkan *Naive Bayes* perlu menghitung probabilitas setiap atribut pada kelas. *Naive Bayes* memiliki persamaan umum sebagai berikut : (Wiratama & Pradnya, 2022).

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)} \qquad (2.1)$$

Keterangan :

X : Data pada *class* yang belum ditemukan,

H : Hipotesis data X menunjukkan suatu *class* spesifik ,

P(H|X) : Probabilitas hipotesis H beralaskan kondisi X (*posteriori prob*.),

P(H) : Probabilitas hipotesis H (*prior prob*.),

P(X|H) : Probabilitas X berdasarkan kondisi dari hipotesis H,

P(X) : Probabilitas dari X

2.6. *Particle Swarm Optimization (PSO)*

Particle Swarm Optimization (PSO) adalah metode yang terinspirasi oleh perilaku alami sekelompok hewan seperti burung, rayap, lebah, atau semut. Algoritma *Particle Swarm Optimization (PSO)* meniru perilaku alami beberapa makhluk tersebut. Ciri-ciri tersebut meliputi kebiasaan yang dilakukan pada aktivitas sehari-hari dan pengaruh individu satu terhadap individu lain di suatu populasi. Kata "partikel" mengacu pada individu, misalnya seperti burung dalam pupulasi burung. Setiap individu atau partikel saling berhubungan dengan kecerdasannya(*intelligence*) tersendiri dan juga dipengaruhi oleh perilaku kelompok lain dalam populasinya. Dengan itu, jika salah satu partikel mendapatkan jalur yang efektif atau lebih pendek ke sumber makannya, maka partikel lain juga akan mengikuti jalur terlepas dari posisi awalnya mereka jauh dari kelompok tersebut (Rizki & Nurlaili, 2021). Berikut tahap-tahapan pada algoritma *Particle Swarm Optimization (PSO)* sebagai berikut : (Sa'diyah et al., 2020).

Langkah 1: Inisialisasi jumlah partikel acak, nilai kecepatan awal partikel = 0

Langkah 2 : Menghitung nilai *fitness* dari setiap partikel menggunakan persamaan 2.2 berikut.

$$f = \frac{\sum_{i=1}^{Nc}\left(\frac{\sum_{j=1}^{Nx} d(xj, ci)}{Nx}\right)}{Nc} \tag{2.2}$$

Keterangan :

Nc = Jumlah *cluster*

Nx = Jumlah data pada *cluster* ke – i

d(xj, ci) = Jarak data dengan *cluster*

Langkah 3 : Perbarui nilai Pbest menggunakan persamaan 2.3 berikut.

$$Pbest = \begin{cases} Pbest_i \ , jika\ F(x_i) \ \geq \ F(Pbest_i) \\ x_i \ , jika\ F(x_i) \ < \ F(Pbest_i) \end{cases} \tag{2.3}$$

Keterangan :

F(x_i) = Nilai *fitness* pada partikel ke – i

F($Pbest_i$) = Nilai *fitness* partikel baik ke – i

Langkah 4 : Perbarui nilai Gbest menggunakan persamaan 2.4 berikut.

$$Gbest_i = MIN(F(Pbest_i), \dots, F(Pbest_n)) \quad (2.4)$$

Langkah 5 : Perbarui nilai kecepatan menggunakan persamaan (2.5) berikut.

$$v_{ij}(t+1) = w.v_{ij}(t) + c_1 r_1 \left[P_{ij}(t) - x_{ij}(t)\right] + c_2 r_2 \left[G_j(t) - x_{ij}(t)\right] \quad (2.5)$$

Keterangan :

W = bobot intertia

$v_{ij}(t)$ = kecepatan itertia ke – i

$c_1\ dan\ c_2$ = nilai konstanta akselerasi

$r_1\ dan\ r_2$ = nilai random dari 0 sampai 1

$x_{ij}(t)$ = partikel ke – i dengan atribut ke – j pada iterasi ke t

$P_{ij}(t)$ = partikel baik ke – i dengan atribut ke – j

$G_j(t)$ = atribut ke – j pada partikel terbaik

Partikel yang bergerak pada *Particle Swarm Optimization (PSO)* dapat meninggalkan ruang pencarian karena rentang kondisi kecepatan yang terlalu jauh sehingga mengakibatkan penurunan efisiensi dan kecepatan konvergensi algoritma. Untuk menghilangkan masalah ini, beberapa batasan dipertimbangkan pada nilai komponen kecepatan. Oleh karena itu, pada setiap iterasi setelah menghitung kecepatan pada Persamaan 2.6, semua komponen dari masing-masing dimensi akan dipertimbangkan dengan membatasi rentang [-Vmax, Vmax] untuk mengurangi probabilitas partikel meninggalkan ruang pencarian.

Langkah 6 : Perbarui nilai posisi dengan rumus yang dapat dilihat pada persamaan 2.6.

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \quad (2.6)$$

Keterangan :

$v_{ij}(t+1)$ = nilai kecepatan pada iterasi ke - $t+1$

$x_{ij}(t)$ = partikel ke – i dengan atribut ke – j pada iterasi ke – t

Langkah 7 : Ulangi langkah 2, 3, dan 4 hingga iterasi *maximum.*

2.7. *K-Fold Cross Validation*

Cross-validation adalah metode teknik data *mining* yang bertujuan untuk mencapai akurasi maksimal saat membagi data menjadi dua subset (data *training* dan data *testing*). Salah satu jenis uji *cross-validation* adalah *K-Fold cross-validation*, yang bertujuan untuk mengevaluasi efisiensi proses suatu metode algoritma dengan membagi data sampel secara acak dan menggabungkan data dengan nilai K dari *K-fold*. Dengan metode *K-Fold Cross Validation*, kumpulan data dibagi secara acak menjadi beberapa bagian. Data partisi tersebut diolah dalam beberapa K percobaan di setiap percobaan, menggunakan data partisi Ke-K sebagai data *testing* dan partisi lain sebagai data *training* (Arisandi et al., 2022).

2.8. *Confusion Matrix*

Confusion matrix adalah metode evaluasi yang digunakan dalam menghitung kinerja atau tingkat akurasi dari proses klasifikasi. *Confusion matrix* memiliki 4 istilah hasil proses klasifikasi yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN) (trivusi, 2022).

1. *True Positive* (TP) = Berarti berapa banyak data yang benar positif, begitu juga modelnya prognosis positif.
2. *True Negative (TN) =* Menunjukkan berapa banyak data sebenarnya negatif, kategori dan model memprediksi yang negatif.
3. *False Positive (FP) =* Berarti berapa banyak data sebenarnya kelas negatif tetapi modelnya prognosis positif.
4. *False Negative (FN) =* Berarti berapa banyak data real kelas positif tetapi modelnya memprediksi yang negatif.

Dari 4 data tersebut dapat digunakan untuk mendapatkan informasi tambahan dalam pengukuran performa sebuah model, seperti : (Saputro & Sari, 2020)

1. *Accuracy* = Total seberapa sering model mengklasifikasi benar. Formula *accuracy* dapat dilihat pada persamaan 2.7.

$$\frac{TP + TN}{Total} \tag{2.7}$$

2. *Precision = Ketika model memprediksi positif, seberapa sering prediksi itu benar. Formula Precision dapat dilihat pada persamaan 2.8.*

$$\frac{TP}{FP + TP} \tag{2.8}$$

3. *Recall (Sensitivity/True Positive Rate) = Ketika kelas aktual positif, seberapa sering model memprediksi secara positif. Formula recall dapat dilihat pada persamaan 2.9.*

$$\frac{TP}{FN + TP} \tag{2.9}$$

4. *F1-Score = Merupakan rata-rata harmonik dari Precision dan Recall. Formula F1-Score dapat dilihat pada persamaan 2.10.*

$$2 * \frac{precision * recall}{precision + recall} \tag{2.10}$$

2.9. *RapidMiner*

RapidMiner adalah perangkat lunak yang bersifat *open source*. *RapidMiner* menggunakan berbagai teknik deskriptif dan prediksi dalam memberikan wawasan terhadap pengguna untuk membuat keputusan yang baik. *RapidMiner* memiliki sekitar 500 operator data *mining*, termasuk operator *input, output,* data *Preprocessing* dan visualisasi. *RapidMiner* ditulis menggunakan bahasa Java agar dapat bekerja di semua sistem operasi (Manullang et al., 2021).