# BAB III
# METODOLOGI PENELITIAN

Bab ini membahas tentang metodologi yang digunakan untuk mengidentifikasi kecurangan (*fraud*) yang meliputi deskripsi masalah, tahap penelitian, dan metode yang digunakan.

## 3.1 Deskripsi Masalah

Mendeteksi *fraud* dalam transaksi perbankan merupakan tantangan yang signifikan dalam industri keuangan. Dalam hal ini, masalah yang dihadapi adalah mendeteksi *fraud* menggunakan *dataset* dari *Bank Simulation* atau BankSim, sebuah simulator pembayaran bank berbasis agen yang menghasilkan data sintetik untuk penelitian deteksi penipuan. BankSim didasarkan pada sampel data transaksi gabungan yang disediakan oleh bank di Spanyol. Tujuan utama BankSim adalah memberikan representasi sintetik dari kegiatan perbankan yang dapat digunakan untuk mempelajari, mengembangkan, dan menguji metode deteksi penipuan yang efektif (Lopez-Rojas & Axelsson, 2014).

Penelitian ini fokus pada *fraud* yang melibatkan kartu kredit. Jenis penipuan ini biasanya dimulai ketika informasi penting pada kartu menjadi terancam: seperti nama akun, nomor kartu kredit, tanggal kadaluarsa, dan kode verifikasi. Informasi ini dapat diperoleh oleh penipu melalui pencurian kartu fisik atau dengan memperoleh pengetahuan tentang data penting yang terhubung dengan akun tersebut.

*Dataset* BankSim memiliki sekumpulan atribut yang dapat mempengaruhi terjadinya *fraud*. Setiap atribut akan diseleksi dengan menggunakan *Particle Swarm Optimization* (PSO), agar terpilih atribut yang paling berpengaruh dalam penentuan *fraud*. Selanjutnya, setiap atribut yang terpilih diklasifikasi dengan *Naive Bayes*. Setelah dilakukan evaluasi dan validasi, maka akan diperoleh prediksi nasabah yang berpotensi melakukan *fraud*.

### 3.2 Data Penelitian

Data yang digunakan dalam penelitian ini adalah data sekunder, yaitu data yang diperoleh dari situs web *Kaggle* (https://www.kaggle.com/) yang diakses pada tanggal 24 Mei 2023. Penelitian ini menggunakan 9 atribut prediktor atau independen dan 1 atribut dependen. Atribut yang digunakan dapat dilihat pada Tabel 3.1 berikut ini.

**Tabel 3.1** Atribut-atribut *Dataset*.

| Atribut | Deskripsi |
|---|---|
| Step | Mewakili hari terjadinya transaksi. |
| Customer | ID unik dari orang yang melakukan transaksi. |
| Age | Umur nasabah yang variabelnya dibagi dalam interval usia, mulai dari 0 hingga 6 dan huruf U=*Unknown.* |
| Gender | Jenis kelamin yang dikategorikan menjadi: E = *Enterprise,* F = *Female,* M = *Male,* U = *Unknown.* |
| ZipCodeOri | Lokasi kode pos asal atau sumber. |
| Merchant | ID unik dari pihak yang menerima transaksi. |
| ZipMarchant | Lokasi kode pos asal atau sumber merchant. |
| Category | Kategori *merchant* yang terdiri dari: *transportation, food, health, wellness and beauty, fashion, bars and restaurants, hyper, sports and toys, tech, home, hotel services, other services, contents, travel, leisure.* |
| Amount | Jumlah transaksi. |
| Fraud | Transaksi yang sah atau tidak sah (*fraud*). |

### 3.3 Pengolahan Data

Untuk mengolah data dilakukan model proses *data mining* (*Data Mining Framework*) terlebih dahulu untuk menghindari data yang tidak konsisten dan tidak lengkap (*missing value*). Model proses *data mining* yang digunakan adalah CRISP-DM. Setelah melewati proses *data mining* tersebut dilakukan pembobotan atribut menggunakan PSO untuk meningkatkan hasil akurasi dari *Naive Bayes*.

Model proses *data mining* dengan menggunakan CRISP-DM terdiri dari lima fase. Berikut adalah penjelasan dari setiap fase pada CRISP-DM:

1. Fase Pemahaman Bisnis (*Business Understanding Phase*)

   Fase ini merupakan tahapan pemahaman atas tujuan dari pengolahan data. Penelitian ini menggunakan *dataset public* berjudul "*Synthetic Data from A Financial Payment System"* dari situs web *Kaggle* (https://www.kaggle.com/). Data tersebut selanjutnya akan diolah agar dapat diklasifikasikan mana transaksi yang sah dan tidak sah (*fraud*).

2. Fase Pemahaman Data (*Data Understanding Phase*)

   Pada fase ini, data mentah yang diperoleh akan ditentukan atribut-atribut yang akan digunakan sebagai prediktor atau hasil. *Dataset* berjumlah 594.643 *record* menggunakan 9 atribut independent dengan tipe atribut *categorical* dan *integer*. Sedangkan atribut hasil atau dependen adalah "fraud".

3. Fase Persiapan Data (*Data Preparation Phase*)

   Pada fase ini dilakukan beberapa teknik untuk mendapatkan kualitas data yang lebih baik karena tidak semua data dan atribut dapat digunakan. Data disebut baik jika tidak ada data yang hilang (*missing value*) dan tipe data untuk setiap atribut sudah sesuai. Berikut adalah tahapan persiapan data:

   a. *Data cleaning*

      Tahapan ini akan dilakukan perbaikan atau pun penghapusan sebagian data yang tidak diperlukan serta, pengecekkan data yang tidak lengkap (*missing value*) dari 10 atribut yang ada.

   *b. Data transformation*

      Data yang diperoleh pada tahapan sebelumnya selanjutnya akan dilakukan transformasi berdasarkan tipe data. Untuk beberapa atribut dengan tipe data *categorical* akan diubah menjadi data *numerical*, dengan cara mengubah setiap nilai dalam kolom menjadi angka yang berurutan. Misalnya, pada atribut "gender", 1 merepresentasikan perempuan (F), 2 merepresentasikan laki-laki (M), 3 merepresentasikan perusahaan (U), dan seterusnya. Pengubahan tipe data dari *categorical* menjadi *numerical* ini bertujuan untuk mempermudah dalam proses pemodelan.

c. *Data reduction*

Tahapan ini dilakukan untuk meningkatkan akurasi dan efisiensi pemrosesan algoritma dengan mengurangi jumlah atribut tanpa mengorbankan kualitas dari hasil yang akan diperoleh. Dalam penelitian ini, digunakan teknik *feature selection* dengan menganalisis korelasi antara variabel independen dan variabel dependen "fraud", serta melakukan pembobotan atribut menggunakan algoritma *Particle Swarm Optimization* (PSO).

d. Penyeimbangan data

Penyeimbangan data dilakukan untuk menghindari bias yang berlebihan terhadap kelas mayoritas dan meningkatkan kemampuan model dalam mengklasifikasikan atau memprediksi kelas minoritas dengan akurasi yang lebih tinggi. Metode yang digunakan untuk mengatasi ketidakseimbangan kelas dalam *dataset* adalah *Synthetic Minority Over-sampling Technique* (SMOTE). Metode ini menghasilkan sampel baru dari kelas minoritas dengan melakukan sampling ulang, sehingga *dataset* menjadi lebih seimbang. Pada penelitian ini yaitu transaksi kartu kredit, sebagian besar transaksi dikategorikan sebagai normal, sementara hanya sedikit transaksi yang tergolong sebagai *fraud*. Ketidakseimbangan tersebut mengakibatkan algoritma klasifikasi menghasilkan akurasi yang lebih tinggi untuk kelas mayoritas, daripada kelas minoritas, hal ini menandakan performa klasifikasi yang buruk (Siringoringo, 2018).

4. Pemodelan (*Modelling Phase*).

Penelitian ini menggunakan algoritma *Naive Bayes Classifier* (NBC) dengan atribut-atribut yang terpilih dari hasil *data reduction* menggunakan PSO. Proses pemodelan dalam penelitian ini menggunakan bantuan *Google Collab* dalam bahasa pemrograman *Python.*

5. Fase Evaluasi dan Validasi (*Evaluation Phase*)

Dalam tahap ini dilakukan validasi dengan dan keakuratan hasil akurasi serta efisiensi algoritma pada model menggunakan beberapa teknik, antara lain

menggunakan *Confussion Matrix*, untuk mengukur akurasi pada model yang digunakan, serta 10-*fold Cross Validation* untuk validasi model.

### 3.4 Algoritma PSO untuk Pembobotan Atribut

Dalam penelitian ini, agar algoritma bekerja dengan lebih efisien dan lebih cepat serta untuk meningkatkan akurasi, maka digunakan *feature selection* dengan pembobotan atribut terlebih dahulu menggunakan algoritma *Particle Swarm Optimization* (PSO). Langkah pertama adalah inisialisasi populasi partikel yang meliputi pemilihan atribut dan pembagian posisi secara kecepatan awal partikel. Kemudian, inisialisasi posisi dan kecepatan awal partikel dengan membangkitkan matriks $\boldsymbol{A} = (a_{ij})$ dan $\boldsymbol{B} = (b_{ij})$ yang elemennya berupa bilangan random pada interval $[0,1]$ berdasarkan banyaknya fitur atau atribut yang telah ditentukan. Banyaknya baris dan kolom dari matriks tersebut masing-masing menyatakan banyaknya jumlah partikel dan jumlah atribut yang diambil. Pada penelitian ini, sebagai inisialisasi populasi, jumlah partikel yang diambil adalah 9 dan jumlah atribut yang dipilih adalah 9. Nilai-nilai pada $\boldsymbol{A}$ juga sekaligus menjadi posisi awal partikel ($X_1^{(1)}$) dan $\boldsymbol{B}$ menjadi kecepatan awal partikel ($V_1^{(1)}$).

Setelah melakukan inisialisasi populasi, tahapan selanjutnya adalah melakukan perhitungan nilai Sigmoid untuk menentukan peluang posisi partikel pada suatu iterasi ke-$t$ dengan menggunakan persamaan berikut (Abualigah & Khader, 2017):

$$S\left(V_i^{(t)}\right) = \frac{1}{(1+e^{-v_i^{(t)}})}. \tag{3.1}$$

$S\left(V_i^{(t)}\right)$ disebut sebagai nilai Sigmoid kecepatan partikel $i$ pada iterasi ke-$t$. Nilai Sigmoid tersebut selanjutnya akan dibandingkan dengan posisi awal setiap partikel ($\boldsymbol{A}$) untuk menentukan kecocokan posisi partikel terhadap nilai Sigmoidnya. Untuk keperluan tersebut, maka didefinisikan matriks kecocokan $\boldsymbol{M}$ sebagai berikut:

$$\boldsymbol{M} = \begin{cases} 1, \text{ jika } X_1^{(1)} \leq S\left(V_1^{(1)}\right) \\ 0, \text{ lainnya.} \end{cases} \tag{3.2}$$

Matriks $\boldsymbol{M}$ ini akan menentukan fitur apa saja yang akan terpilih pada setiap partikel.

Tahapan selanjutnya adalah perhitungan nilai $P_{best}$ yang akan digunakan untuk menentukan posisi terbaik yang dimiliki oleh setiap partikel. Nilai $P_{best}$ mencerminkan seberapa akurat atau valid prediksi posisi partikel pada dimensi tertentu. Semakin tinggi nilai $P_{best}$, semakin baik atau lebih dekat posisi partikel tersebut dengan minimum lokal dari fungsi yang sedang dioptimalkan (Kamaruddin & Ravi, 2016). Penentuan nilai $P_{best}$ digunakan untuk mengevaluasi setiap solusi yang dihasilkan oleh metode ini. Pada setiap iterasi, nilai $P_{best}$ dari setiap solusi dihitung untuk menentukan apakah ada perbaikan dalam solusi tersebut atau tidak. Perhitungan nilai $P_{best}$ dilakukan dengan menggunakan persamaan berikut:

$$P_{best} = \frac{1}{n}\sum \left| \frac{V_i^{(t)} - S(V_i^{(t)})}{V_i^{(t)}} \right|. \quad (3.3)$$

Solusi dengan nilai $P_{best}$ tertinggi akan diambil sebagai solusi terbaik pada iterasi ke satu, dan disimbolkan dengan $G_{best}$. Setelah itu dilanjutkan dengan iterasi ke dua.

Langkah awal pada iterasi ke dua adalah menentukan parameter bobot inersia ($\omega$) yang akan digunakan untuk mengurangi kecepatan partikel dengan menggunakan persamaan berikut:

$$\omega^{(t)} = \omega_{max} - \left(\frac{\omega_{max} - \omega_{min}}{t_{max}}\right) t. \quad (3.4)$$

di mana:

$t$ = iterasi.

$\omega^{(t)}$ = Nilai bobot inersia pada iterasi ke$-t$.

$\omega_{max}$ = Nilai bobot inersia maksimum.

$\omega_{min}$ = Nilai bobot inersia minimum.

$t_{max}$ = iterasi maksimum.

Nilai bobot inersia maksimum dan minimum serta banyaknya iterasi maksimum merupakan nilai parameter yang harus ditetapkan terlebih dahulu sebelum memulai algoritma PSO. Selanjutnya, nilai bobot inersia tersebut digunakan untuk melakukan *update* kecepatan partikel dengan menggunakan persamaan berikut:

$$V_i^{(t)} = \omega\, V_i^{(t-1)} + c_1 \boldsymbol{A}\left(P_{best} - X_i^{(t-1)}\right) + c_2 \boldsymbol{B}\left(G_{best} - X_i^{(t-1)}\right). \quad (3.5)$$

Formula di atas adalah formula modifikasi usulan dari (Shi & Eberhart, 1998) digunakan untuk mengurang kecepatan partikel secara bertahap agar kecepatan partikel tidak berubah terlalu cepat sehingga melampaui nilai minimum yang diinginkan.

Selanjut dilakukan *update* posisi partikel dengan menggunakan persamaan berikut:

$$X_i^{(t)} = V_i^{(t)} + X_i^{(t-1)}. \quad (3.6)$$

Setelah diperoleh kecepatan dan posisi partikel yang baru maka langkah selanjutnya adalah menghitung nilai Sigmoid, menghitung matriks kecocokan, menentukan nilai $P_{best}$ dan $G_{best}$ sebelum masuk ke iterasi selanjutnya. Proses ini diulang sampai mencapai iterasi $t_{max}$.

Hasil dari proses PSO akan dijadikan sebagai seleksi atribut. Untuk atribut yang menghasilkan bobot 0 dapat dihilangkan, karena atribut tersebut tidak memiliki korelasi atau tidak berpengaruh. Sedangkan untuk atribut yang bernilai 1 akan digunakan untuk proses klasifikasi. Setelah diperoleh atribut terpilih, maka dilanjutkan dengan pembuatan model dengan menggunakan *Naive Bayes Classifier*. Proses pembangunan model tersebut akan dijelaskan di sub bab selanjutnya.

### 3.5 *Naive Bayes Classifier* (NBC)

*Naive Bayes Classifier* menggunakan metode probabilistik untuk memprediksi kelas untuk setiap kumpulan data. Pembuatan model *Naive Bayes Classifier* dilakukan pada *dataset* yang terdiri dari atribut-atribut hasil pembobotan dengan PSO. Pada saat membuat model *Naive Bayes Classifier* terlebih dahulu kita harus mencari nilai probabilitas hipotesis untuk masing-masing kelas $P(H)$. Hipotesis yang ada yaitu *fraud* dan tidak *fraud*. *Data training* yang digunakan yaitu 594.643 data, 587.443 data dengan pembayaran normal atau tidak *fraud* dan 7200 data adalah *fraud*, perhitungan probabilitas seperti di bawah ini:

$$P(not\ fraud) = 587.443 : 594.643 = 0{,}987$$

$$P(fraud) = 7200 : 594.643 = 0{,}012$$

Setelah probabilitas untuk tiap hipotesis diketahui langkah selanjutnya adalah menghitung probabilitas kondisi tertentu (probabilitas X) berdasarkan

probabilitas tiap hipotesis (probabilitas H) atau dinamakan probabilitas prior untuk atribut setiap kelas dengan menggunakan *Naive Bayes*.

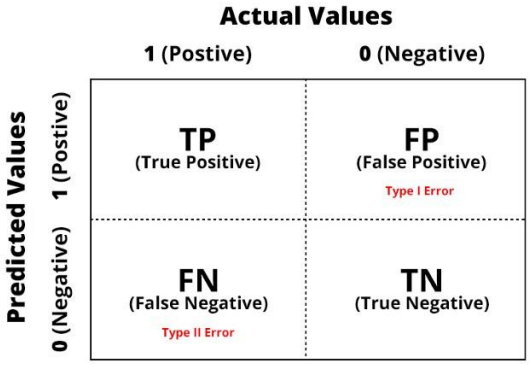
Langkah kerja khusus *Naive Bayes* adalah sebagai berikut (Li, Ding, & Li, 2014). Misal $T$ adalah kumpulan *data training sample*. Setiap sampel memiliki atribut/label kategori. Kumpulan sampel memiliki $m$ kelas: $C_1, C_2, \dots, C_m$. Setiap sampel diwakili oleh $n$-dimensi $X = \{x_1, x_2, \dots, x_n\}$, dan setiap vektor menggambarkan $n$ atribut $A_1, A_2, \dots, A_n$.

1. Diberikan $X$ sederhana, pengklasifikasi akan memprediksi bahwa $X$ memiliki probabilitas posterior kelas tertinggi jika dan hanya jika $P(C_i|X) > P(C_j|X), 1 \leq i, j \leq m,$ $X$ diprediksi termasuk kelas $C$. Sehingga menurut Teorema Bayes $P(C_i|X) = \frac{P(X|C_i)\,.\,P(C_i)}{P(X)}$. Karena $P(X)$ sama untuk semua kelas, hanya perlu mencari $P(X|C_i).P(C_i)$ terbesar. Probabilitas sebelumnya dari kelas $C_i$ dapat dihitung dengan rumus $P(C_i) = \frac{s_i}{s}$, di mana $s_i$ adalah jumlah *training sample* dari kelas $C$ dan $s$ adalah jumlah total *training sample*.
2. Jika *dataset* memiliki banyak atribut, beban kerja untuk menghitung $P(X|C_i)$ menjadi sangat tinggi. Untuk mengurangi *overhead* komputasi dari $P(X|C_i)$, diasumsukan bahwa dalam kondisi tertentu nilai karakteristik atribut tidak bergantung satu sama lain. Dirumuskan dalam matematika sebagai berikut:
$$P(X|C_i) = \prod_{k=1}^{n} P(X_k|C_i). \quad (3.7)$$
3. Probabilitas $P(x_1|C_i), P(x_2|C_i), \dots, P(x_n|C_i)$ dapat dihitung dari *dataset training*. $x_k$ mengacu pada atribut $A_k$ dari sampel $X$.
4. Untuk setiap kelas, hitunglah $P(X|C_i)P(C_i)$ jika dan hanya jika $P(X|C_i)P(C_i)$ maksimum, *classifier prediction sample* dari $X$ merupakan kelas $C_i$.

Hasil dari perhitungan *Naive Bayes Classifier* memberikan probabilitas untuk setiap kelas yang akan dievaluasi dan divalidasi dengan metode *Confusion Matrix* dan 10-*fold Cross Validation*. Proses evaluasi dan validasi tersebut akan dijelaskan pada sub bab selanjutnya.

### 3.6 Validasi dan Evaluasi

Dalam penelitian ini, *Confusion Matrix* dan 10-*fold Cross Validation* digunakan untuk mengukur akurasi dan validitas model yang digunakan. *Confusion Matrix* merupakan salah satu metode berbentuk tabel matriks yang dapat digunakan untuk mengukur kinerja metode klasifikasi. Pada dasarnya *Confusion Matrix* mengandung informasi yang membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang seharusnya (Prasetyo, 2012). *Confusion Matrix* dapat disajikan seperti pada Gambar 3.1 berikut ini.




**Gambar 3.1** *Confusion Matrix*.

*Confusion Matrix* akan menunjukkan hasil dari proses klasifikasi, yang terdiri dari:

1. *True Positive* (TP), yang merupakan jumlah data aktual yang benar diprediksi benar.
2. *True Negative* (TN), yang merupakan jumlah data aktual yang salah diprediksi salah.
3. *False Positive* (FP), yang merupakan jumlah data aktual yang benar diprediksi salah.
4. *False Negative* (FN), yang merupakan jumlah data aktual yang salah diprediksi benar.

Keempat klasifikasi tersebut akan digunakan sebagai dasar untuk metrik evaluasi. Metrik evaluasi akan dihitung menggunakan rumus yang telah ditentukan sebagai berikut:

$$\text{Akurasi} = \frac{TP+TN}{TP+FN+FP+TN} \times 100\%.$$

$$\text{Presisi} = \frac{TP}{TP+FP} \times 100\%.$$

$$\textit{Recall} = \frac{TP}{TP+FN} \times 100\%.$$

Nilai akurasi menggambarkan seberapa akurat model dapat mengklasifikasikan data dengan benar. Akurasi merupakan tingkat kedekatan nilai prediksi dengan nilai aktual (sebenarnya). Nilai presisi menggambarkan tingkat keakuratan antara data yang diminta dengan hasil prediksi yang diberikan oleh model, serta nilai *recall* menunjukkan berapa persen data kategori positif yang terklasifikasikan dengan benar oleh model (Nugroho, 2019).

Setelah itu dilakukan *k-fold Cross Validation* untuk mengestimasi kesalahan prediksi dalam mengevaluasi kinerja model. Data dibagi menjadi himpunan bagian $k$ berjumlah hampir sama. Model dalam klasifikasi dilatih dan diuji sebanyak $k$, dengan $k = 10$. Di setiap pengulangan, salah satu himpunan bagian akan digunakan sebagai *data training* dan *data testing*. Langkah-langkah dari 10-*fold Cross Validation* yaitu:

1. Total data dibagi menjadi *k* bagian.
2. *Fold* ke-1 adalah ketika bagian ke-1 menjadi data uji (*testing data*) dan sisanya menjadi data latih (*training data*). Kemudian, hitung akurasi atau kesamaan atau kedekatan suatu hasil pengukuran dengan angka atau data yang sebenarnya berdasarkan porsi data tersebut. Perhitungan akurasi tersebut menggunakan persamaan sebagai berikut

$$akurasi = \frac{\sum data\ uji\ benar\ klasifikasi}{\sum total\ data\ uji} \times 100.$$

3. *Fold* ke-2 adalah ketika bagian ke-2 menjadi data uji (*testing data*) dan sisanya menjadi data latih (*training data*). Kemudian hitung akurasi berdasarkan porsi data tersebut.
4. Demikian seterusnya hingga mencapai *fold* ke-10. Kemudian hitung rata-rata akurasi dari 10 buah akurasi di atas. Rata-rata akurasi ini menjadi akurasi final.

### 3.7 Contoh Kasus

Pada sub bab ini akan diberikan contoh kasus sederhana mulai dari langkah-langkah algoritma *Particle Swarm Optimization* (PSO) sampai dengan klasifikasi

menggunakan *Naive Bayes Classifier.* Misal terdapat *dataset* transaksi yang terdiri dari 10 atribut dengan "Risk" sebagai variabel dependen dan 9 atribut lainnya sebagai varibel independen yang disajikan dalam Tabel 3.2 berikut ini.

**Tabel 3.2** *Dataset* Transaksi.

| ***Age*** | ***Sex*** | ***Job*** | ***Housing*** | ***Saving accounts*** | ***Checking account*** | ***Credit amount*** | ***Duration*** | ***Purpose*** | ***Risk*** |
|---|---|---|---|---|---|---|---|---|---|
| 67 | 1 | 2 | 2 | 0 | 1 | 1169 | 6 | 4 | 1 |
| 22 | 0 | 2 | 2 | 1 | 2 | 5951 | 48 | 4 | 0 |
| 49 | 1 | 1 | 2 | 1 | 2 | 2096 | 12 | 2 | 1 |
| 45 | 1 | 2 | 0 | 1 | 1 | 7882 | 42 | 3 | 1 |
| 53 | 1 | 2 | 0 | 1 | 1 | 4870 | 24 | 1 | 0 |

Berdasarkan Tabel 3.2, inisialisasi populasi jumlah partikel yang diambil adalah 5 dan jumlah fitur yang dipilih ada 9, yaitu "Age", "Sex", "Job", "Housing", "Saving accounts", "Checking account", "Credit amount", "Duration", dan "Purpose".

Berikut diberikan contoh langkah-langkah algoritma *Particle Swarm Optimization* (PSO) untuk *feature selection*.

**Langkah 1:** Membangkitkan matriks $\boldsymbol{A} = (a_{ij})$ dan $\boldsymbol{B} = (b_{ij})$ yang elemennya berupa bilangan random pada interval [0,1]. Elemen-elemen dari matriks $\boldsymbol{A}$ dan $\boldsymbol{B}$ dituliskan pada Tabel 3.3 dan Tabel 3.4 berikut ini.

**Tabel 3.3** Matriks $\boldsymbol{A}$.

| **Partikel** | ***A*** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | 0,6937 | 0,6091 | 0,6081 | 0,4246 | 0,8099 | 0,5889 | 0,7982 | 0,3976 | 0,1191 |
| $P_2$ | 0,2202 | 0,3413 | 0,0495 | 0,7231 | 0,8772 | 0,3952 | 0,2713 | 0,8824 | 0,4647 |
| $P_3$ | 0,8176 | 0,6510 | 0,4734 | 0,0863 | 0,4582 | 0,0655 | 0,3480 | 0,7730 | 0,0742 |
| $P_4$ | 0,8665 | 0,4014 | 0,4742 | 0,5877 | 0,9439 | 0,1872 | 0,0768 | 0,3128 | 0,3998 |
| $P_5$ | 0,5298 | 0,7668 | 0,8058 | 0,8000 | 0,2257 | 0,8514 | 0,0489 | 0,8751 | 0,2538 |

**Tabel 3.4** Matriks $\boldsymbol{B}$.

| **Partikel** | $\boldsymbol{B}$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | 0,4257 | 0,2744 | 0,1913 | 0,4390 | 0,1860 | 0,2889 | 0,7679 | 0,0488 | 0,2695 |
| $P_2$ | 0,7523 | 0,6746 | 0,6643 | 0,5692 | 0,8139 | 0,3911 | 0,7373 | 0,4473 | 0,6997 |
| $P_3$ | 0,9985 | 0,7502 | 0,7742 | 0,5008 | 0,8175 | 0,4557 | 0,4224 | 0,3150 | 0,9235 |
| $P_4$ | 0,2368 | 0,6767 | 0,9874 | 0,0279 | 0,8349 | 0,4958 | 0,1866 | 0,9131 | 0,3109 |
| $P_5$ | 0,7072 | 0,2440 | 0,6529 | 0,7060 | 0,6721 | 0,2349 | 0,6955 | 0,4417 | 0,3464 |

**Langkah 2:** Hitung nilai Sigmoid untuk setiap partikel dengan cara memasukan nilai kecepatan awal partikel **($\boldsymbol{B}$)** menggunakan Persamaan (3.1)**.** Hasil nilai Sigmoid untuk iterasi pertama $S\left(V_1^{(1)}\right)$ dapat dilihat pada Tabel 3.5 berikut ini.

**Tabel 3.5** Nilai Sigmoid.

| **Partikel** | $\boldsymbol{S\left(V_1^{(1)}\right)}$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | 0,6049 | 0,5682 | 0,5477 | 0,6080 | 0,5464 | 0,5717 | 0,6831 | 0,5122 | 0,5670 |
| $P_2$ | 0,6797 | 0,6625 | 0,6602 | 0,6386 | 0,6929 | 0,5965 | 0,6764 | 0,6100 | 0,6681 |
| $P_3$ | 0,7308 | 0,6792 | 0,6844 | 0,6226 | 0,6937 | 0,6120 | 0,6041 | 0,5781 | 0,7158 |
| $P_4$ | 0,5589 | 0,6630 | 0,7286 | 0,5070 | 0,6974 | 0,6215 | 0,5465 | 0,7136 | 0,5771 |
| $P_5$ | 0,6698 | 0,5607 | 0,6577 | 0,6695 | 0,6620 | 0,5584 | 0,6672 | 0,6087 | 0,5857 |

Selanjutnya, bandingkan posisi awal setiap partikel **($\boldsymbol{A}$**) dengan nilai Sigmoid untuk memperoleh matriks kecocokan $\boldsymbol{M}$ dengan ketentuan yang ada pada Persamaan (3.2). Hasil matriks kecocokan $\boldsymbol{M}$ dapat dilihat pada Tabel 3.6 berikut ini.

**Tabel 3.6** Hasil Perbandingan $\boldsymbol{A}$ dengan Nilai Sigmoid.

| $\boldsymbol{j}$ / $\boldsymbol{i}$ | $\boldsymbol{f_1}$ | $\boldsymbol{f_2}$ | $\boldsymbol{f_3}$ | $\boldsymbol{f_4}$ | $\boldsymbol{f_5}$ | $\boldsymbol{f_6}$ | $\boldsymbol{f_7}$ | $\boldsymbol{f_8}$ | $\boldsymbol{f_9}$ |
|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| $P_2$ | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| $P_3$ | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| $P_4$ | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| $P_5$ | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |

dengan 1=terpilih dan 0=tidak terpilih.

Berdasarkan nilai pada Tabel 3.6, dapat dilihat bahwa:

a. Pada $P_1$ fitur yang terpilih adalah $f_4$ (*Housing)*, $f_8$ (*Duration)*, dan $f_9$ (*Purpose*).
b. Pada $P_2$ fitur yang terpilih adalah $f_1$ (*Age*), $f_2$ (*Sex*), $f_3$ (*Job*), $f_6$ (*Checking account*), $f_7$ (*Credit amount*), dan $f_9$ (*Purpose*).
c. Pada $P_3$ fitur yang terpilih adalah $f_2$ (*Duration*), $f_3$ (*Housing*), $f_4$ (*Housing*) $f_5$ (*Saving accounts*), $f_6$ (*Duration*), $f_7$ (*Credit amount*), dan $f_9$ (*Purpose*).
d. Pada $P_4$ fitur yang terpilih adalah $f_2$ (*Sex*), $f_3$ (*Job*), $f_6$ (*Duration*), $f_7$ (*Credit amount*), $f_8$ (*Duration*), dan $f_9$ (*Purpose*).
e. Pada $P_5$ fitur yang terpilih adalah $f_1$ (*Age*), $f_5$ (*Saving accounts*), $f_7$ (*Credit amount*), dan $f_9$ (*Purpose*).

**Langkah 3:** Hitung nilai $P_{best}$ menggunakan Persamaan (3.3). Hasil perhitungan $P_{best}$ dapat dilihat pada Tabel 3.7 berikut ini.

**Tabel 3.7** Hasil $P_{best}$.

| Partikel | $\boldsymbol{P_{best}}$ |
|---|---|
| $P_1$ | 0,0802 |
| $P_2$ | 0,0024 |
| $P_3$ | 0,0006 |
| $P_4$ | 0,0202 |
| $P_5$ | 0,02 |

Pada Tabel 3.7 dapat dilihat solusi dengan nilai tertinggi adalah $P_1$ sehingga, dapat dipilih 0,0802 sebagai nilai dari $G_{best}$. Berdasarkan Tabel 3.6 fitur yang terpilih pada iterasi pertama adalah fitur yang dimiliki oleh $P_1$ yaitu $f_4$ (*Housing)*, $f_8$ (*Duration)*, dan $f_9$ (*Purpose*).

**Langkah 4:** Selanjutnya, untuk langkah awal iterasi kedua adalah menentukan parameter bobot inersia ($\omega$) dengan menggunakan Persamaan (3.4). Misalkan dipilih $\omega_{max} = 0.9$, $\omega_{min} = 0.4$, $t_{max} = 10$, dengan $t = 2$. Maka, diperoleh:

$$\omega^{(2)} = 0.9 - \left(\frac{0.9 - 0.4}{10}\right) 2$$

$$\omega^{(2)} = 0.8$$

Kemudian, dilakukan *update* kecepatan partikel dengan menggunakan bobot inersia $\omega^{(2)}$, tabel matriks $\boldsymbol{A}$ dan $\boldsymbol{B}$, nilai $P_{best}$ dan $G_{best}$ pada iterasi pertama dan parameter $c_1, c_2 = 1$. Hasil kecepatan partikel dengan menggunakan Persamaan (3.5) dapat dilihat pada Tabel 3.8 berikut ini.

**Tabel 3.8** Kecepatan Partikel Iterasi ke Dua.

| **Partikel** | $\boldsymbol{V_2^{(2)}}$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | -0,3292 | -0,2345 | -0,2569 | 0,0668 | -0,5629 | -0,2022 | -0,4864 | -0,0959 | 0,2064 |
| $P_2$ | 0,4594 | 0,2573 | 0,5594 | -0,4249 | -0,7547 | 0,0394 | 0,3865 | -0,7727 | 0,0854 |
| $P_3$ | -0,5899 | -0,2397 | 0,1031 | 0,3978 | 0,1479 | 0,3739 | 0,1105 | -0,5581 | 0,7529 |
| $P_4$ | -0,7247 | 0,1821 | 0,2015 | -0,3237 | -0,9104 | 0,3202 | 0,1486 | 0,4410 | 0,0031 |
| $P_5$ | -0,0154 | -0,5469 | -0,5805 | -0,5625 | 0,4019 | -0,7038 | 0,5869 | -0,7457 | 0,1610 |

**Langkah 5:** *Update* posisi partikel ($X_2^{(2)}$) menggunakan Persamaan (3.6) sehinggga, diperoleh posisi partikel baru pada iterasi ke dua seperti pada Tabel 3.9 berikut ini.

**Tabel 3.9** Posisi Partikel Iterasi ke Dua.

| **Partikel** | $\boldsymbol{X_2^{(2)}}$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $P_1$ | 0,3644 | 0,3747 | 0,3512 | 0,4914 | 0,2470 | 0,3867 | 0,3117 | 0,3017 | 0,3254 |
| $P_2$ | 0,6796 | 0,5986 | 0,6089 | 0,2983 | 0,1225 | 0,4346 | 0,6578 | 0,1096 | 0,5501 |
| $P_3$ | 0,2277 | 0,4113 | 0,5765 | 0,4841 | 0,6061 | 0,4394 | 0,4585 | 0,2149 | 0,8271 |
| $P_4$ | 0,1418 | 0,5834 | 0,6757 | 0,2640 | 0,0335 | 0,5074 | 0,2254 | 0,7538 | 0,4030 |
| $P_5$ | 0,5144 | 0,2199 | 0,2252 | 0,2375 | 0,6276 | 0,1476 | 0,6358 | 0,1294 | 0,4148 |

**Langkah 6:** Ulangi Langkah 2 sampai Langkah 5 hingga mencapai iterasi $t_{max}$.

Setelah dilakukan seleksi fitur menggunakan PSO selanjutnya fitur-fitur yang terpilih akan diklasifikasi menggunakan *Naive Bayes*. Misalkan fitur-fitur yang terpilih adalah "Sex", "Job", "Checking account", "Credit amount", dan "Purpose". *Dataset* yang terseleksi tersebut disajikan pada Tabel 3.10 berikut ini.

**Tabel 3.10** *Dataset* Transaksi Setelah Diseleksi.

| ***Sex*** | ***Job*** | ***Credit amount*** | ***Purpose*** | ***Risk*** |
|---|---|---|---|---|
| 1 | 2 | 1169 | 4 | 1 |
| 0 | 2 | 5951 | 4 | 0 |
| 1 | 1 | 2096 | 2 | 1 |
| 1 | 2 | 7882 | 3 | 1 |
| 1 | 2 | 4870 | 1 | 0 |

Hasil pada Tabel 3.10 menggambarkan faktor-faktor yang mempengaruhi *Risk* dari transaksi kartu kredit, di mana 1 merepresentasikan *Risk Good* dan 0 merepresentasikan *Risk Bad*. Selanjutnya, *dataset* pada Tabel 3.10 dibagi menjadi dua bagian, yaitu *Risk* sebagai atribut dependen dan 4 atribut lainnya sebagai fitur prediktor/independen.

Misalkan terdapat dua kelas $C_1$ dan $C_2$, yaitu untuk *Risk Good* dan *Risk Bad*. Langkah awal yang perlu dilakukan adalah mencari nilai probabilitas hipotesis $P(H)$ untuk masing-masing kelas. Perhitungan probabilitas seperti berikut:

$$P(Good) = \frac{s_i}{s} = \frac{3}{5}$$

$$P(Bad) = \frac{s_i}{s} = \frac{2}{5}$$

dengan $s_i$ adalah jumlah *training sample* dari masing-masing kelas $C_i$ dan $s$ adalah jumlah total *training sample*. Kemudian, dengan menggunakan formula *Naive Bayes*, probabilitas masing-masing fitur dependen dapat dihitung yang disajikan pada Tabel 3.11 berikut ini.

**Tabel 3.11** Probabilitas Setiap Atribut.

| **Atribut** | **Nilai** | ***Risk*** | | **Probabilitas** | |
|---|---|---|---|---|---|
| | | ***Good*** | ***Bad*** | ***Good*** | ***Bad*** |
| *Sex* | 1-*female* | 3 | 1 | $^{3}/_{3}$ | $1/2$ |
| | 0-*male* | 0 | 1 | 0 | $1/2$ |
| *Job* | 1-*unskilled* | 1 | 0 | $1/3$ | 0 |
| | 2-*skilled* | 2 | 2 | $2/3$ | $2/2$ |
| *Credit amount* | 1000-3000 | 2 | 0 | $2/3$ | 0 |
| | 3001-5000 | 0 | 1 | 0 | $1/2$ |
| | 5001-7000 | 0 | 1 | 0 | $1/2$ |
| | $> 7000$ | 1 | 0 | $1/3$ | 0 |
| *Purpose* | 1-*car* | 0 | 1 | 0 | $1/2$ |
| | 2-*TV* | 1 | 0 | $1/3$ | 0 |
| | 3-*repairs* | 1 | 0 | $1/3$ | 0 |
| | 4-e*ducation* | 1 | 1 | $1/3$ | $1/2$ |

Setelah menghitung probabilitas untuk setiap faktor, dilakukan pengambilan keputusan berdasarkan nilai probabilitas kelas "*Good*" dan "*Bad*" untuk setiap faktor dengan ketentuan *Sex* adalah *Female*, *Job* adalah *unskilled*, *Credit amount* yang berjumlah >7000, dan *Purpose* adalah TV. Misal diberikan *X* sebagai berikut:

$X = (Female, Unskilled, > 7000, TV)$, akan dihitung $P(C_i|X)$ untuk setiap kelas $C_1$ dan $C_2$ menggunakan Teorema Bayes $P(C_i|X) = \frac{P(X|C_i)\,.\,P(C_i)}{P(X)}$. Karena $P(X)$ sama untuk semua kelas sehingga, hanya perlu mencari $P(X|C_i).P(C_i)$ terbesar. Untuk perhitungan dari $P(X|C_i)$ digunakan Persamaan (3.7), sehingga diperoleh:

$$P(X|Good).P(Good) = (\frac{3}{3})(\frac{1}{3})(\frac{1}{3})(\frac{1}{3})(\frac{3}{5}) = 0{,}022$$

$$P(X|Bad).P(Bad) = (\frac{1}{2})(\frac{2}{2})(\frac{1}{2})(\frac{1}{2})(\frac{2}{5}) = 0{,}05$$

Karena $P(X|Bad) = 0{,}05 > P(X|Good) = 0{,}022$, maka *classifier prediction sample* dari $X$ merupakan kelas $C_2$ atau bisa direpresentasikan bahwa transaksi kartu kredit tersebut dinyatakan sebagai *Risk Bad.*