BAB 2
LANDASAN TEORI

2.1 Manajemen Keuangan Pribadi

Manajemen keuangan pribadi dapat dipahami sebagai proses pengelolaan arus kas individu untuk mencapai stabilitas finansial jangka pendek dan kesejahteraan finansial jangka panjang (*financial well-being*). Mengelola keuangan pribadi sering dianggap sebagai salah satu aspek paling menantang dalam kehidupan sehari-hari. Hal ini membutuhkan perhatian yang teliti terhadap berbagai detail, pemahaman yang mendalam mengenai kondisi keuangan diri sendiri, serta pendekatan yang terorganisasi dengan baik dalam menyusun anggaran dan mengatur pengeluaran. Meskipun demikian, pengelolaan keuangan pribadi yang efektif sangat penting, tidak hanya untuk menjaga kestabilan finansial, tetapi juga untuk mencapai tujuan keuangan jangka panjang, seperti menabung untuk masa pensiun, membeli rumah, atau membiayai pendidikan anak [17].

Perilaku keuangan individu bersifat heterogen dan sangat dipengaruhi oleh karakteristik sosial-ekonomi, seperti usia, status pernikahan, jumlah tanggungan, serta tingkat pendapatan. Kondisi ini mendorong penerapan pendekatan *financial profiling* untuk mengidentifikasi pola pengeluaran berdasarkan segmentasi demografis. Penelitian terdahulu menunjukkan adanya perbedaan signifikan dalam struktur konsumsi antar kelompok demografis, sehingga model penganggaran yang bersifat umum kerap tidak mampu mencerminkan kebutuhan spesifik masing-masing individu [1]. Di sisi lain, indikator ekonomi eksternal seperti perbedaan biaya hidup dan standar kebutuhan antar wilayah juga memengaruhi kapasitas tabungan individu secara substansial. Penelitian berbasis machine learning di negara berkembang turut mengonfirmasi bahwa segmentasi perilaku finansial yang dikombinasikan dengan pendekatan *explainable machine learning* dapat mengungkap pola literasi dan pengelolaan keuangan yang berbeda-beda antar kelompok demografis, sehingga intervensi finansial yang ditargetkan menjadi jauh lebih efektif dibandingkan pendekatan yang bersifat seragam [11]. Hal ini memperkuat urgensi penerapan pendekatan *cost-of-living adjusted budgeting*, yang menekankan bahwa proporsi alokasi keuangan tidak dapat dilepaskan dari konteks geografis dan daya beli di wilayah yang bersangkutan.

Implementasi konkret dari prinsip *cost-of-living adjusted* dalam penelitian

ini dilakukan dengan mempertahankan seluruh nilai finansial pada dataset dalam mata uang aslinya, yaitu Rupee India (INR), tanpa dikonversi ke mata uang negara lain. Pendekatan ini didasari oleh pertimbangan bahwa nilai tukar mata uang semata tidak merepresentasikan perbedaan daya beli maupun struktur biaya hidup yang sesungguhnya antarnegara, sehingga konversi nilai tukar justru berisiko menghasilkan interpretasi yang keliru apabila diterapkan lintas konteks ekonomi yang berbeda, meskipun kedua negara sama-sama tergolong sebagai negara berkembang. Oleh karena itu, penelitian ini memosisikan India sebagai studi kasus tunggal yang representatif terhadap karakteristik negara berkembang, di mana seluruh pola pendapatan, pengeluaran, dan alokasi anggaran dianalisis sepenuhnya dalam konteks ekonomi domestiknya sendiri, tanpa dimaksudkan untuk digeneralisasikan secara langsung terhadap kondisi finansial negara berkembang lain.

2.2 K-Means Clustering

K-Means Clustering merupakan salah satu algoritma *unsupervised learning* yang digunakan untuk mengelompokkan data ke dalam sejumlah klaster berdasarkan kemiripan karakteristik tertentu. Dalam penelitian ini, *K-Means* berfungsi sebagai algoritma klasifikasi yang membagi seluruh data pengguna ke dalam segmen-segmen berbasis kemiripan profil finansial dan demografis. Setiap segmen yang terbentuk kemudian digunakan sebagai basis pelatihan model FNN secara terpisah, sehingga prediksi alokasi keuangan yang dihasilkan bersifat spesifik terhadap karakteristik tiap kelompok pengguna.

Metode ini dimulai dengan memilih K objek secara acak sebagai pusat klaster awal. Kemudian, jarak antara setiap objek dan pusat klaster tersebut dihitung, dan setiap objek ditempatkan ke klaster yang memiliki pusat terdekat [18]. Pendekatan ini telah banyak diterapkan dalam konteks segmentasi perilaku pengeluaran dan keuangan individu, di mana *K-Means* terbukti mampu membentuk kelompok pelanggan yang bermakna berdasarkan data pembelian dan karakteristik demografis secara efisien [10]. Pemilihan centroid awal dalam penelitian ini dilakukan menggunakan metode *K-Means++*, yang dirancang untuk mengatasi kelemahan inisialisasi acak murni pada *K-Means* klasik. Pada inisialisasi acak murni, *centroid* awal dipilih sepenuhnya secara acak dari data, sehingga berisiko menghasilkan pengelompokan yang buruk atau konvergensi yang lambat. *K-Means++* mengatasi hal ini dengan memilih *centroid* secara bertahap berdasarkan

distribusi probabilitas yang memprioritaskan titik data yang jauh dari *centroid* yang sudah dipilih sebelumnya. Secara matematis, probabilitas suatu titik data x_i terpilih sebagai *centroid* berikutnya dirumuskan sebagai:

$$P(x_i) = \frac{D(x_i)^2}{\sum_{x \in X} D(x)^2} \tag{2.1}$$

di mana $D(x_i)$ menyatakan jarak minimum dari titik x_i ke *centroid* terdekat yang telah dipilih sebelumnya. Dengan mekanisme ini, *centroid* awal tersebar lebih merata di ruang fitur sehingga proses iterasi *K-Means* cenderung lebih stabil dan menghasilkan solusi akhir yang lebih optimal dibandingkan inisialisasi acak.

Selain itu, *K-Means* berupaya meminimalkan variasi intra-klaster yang direpresentasikan melalui fungsi objektif *Within-Cluster Sum of Squares* (WCSS) [19, 20]. Fungsi ini mengukur total kuadrat jarak antara setiap data dengan *centroid* klasternya dan dirumuskan sebagai:

$$WCSS = \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - \mu_j\|^2 \tag{2.2}$$

di mana C_j adalah klaster ke-j dan μ_j adalah *centroid* dari klaster tersebut. Proses pengelompokan dilakukan secara iteratif dengan meminimalkan nilai WCSS hingga tercapai konvergensi, yaitu ketika tidak terjadi perubahan signifikan pada posisi *centroid* atau keanggotaan klaster. Nilai WCSS yang lebih rendah mengindikasikan bahwa titik-titik data berada lebih dekat dengan centroid cluster-nya masing-masing, sehingga mencerminkan kualitas pengelompokan yang lebih baik [21].

Pengukuran kedekatan antara data dan *centroid* dalam penelitian ini menggunakan *Euclidean Distance*, yang sesuai untuk data numerik berdimensi d. Jarak antara suatu titik data $x_i = (x_i1, x_i2, ..., x_id)$ dan *centroid* $\mu_j = (\mu_j1, \mu_j2, ..., \mu_jd)$ didefinisikan sebagai:

$$d(x_i, \mu_j) = \sqrt{\sum_{l=1}^{d} (x_{il} - \mu_{jl})^2} \tag{2.3}$$

Setelah seluruh data dialokasikan ke klaster terdekat, *centroid* diperbarui menggunakan rata-rata aritmetika dari seluruh anggota klaster. Proses pembaruan

centroid dirumuskan sebagai:

$$\mu_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i \tag{2.4}$$

dengan $|C_j|$ menyatakan jumlah anggota pada *cluster* ke-j. Iterasi antara tahap perhitungan jarak dan pembaruan *centroid* dilakukan hingga fungsi objektif mencapai nilai minimum lokal, yang menandakan struktur klaster telah stabil.

Untuk mengevaluasi kualitas pengelompokan dan mendukung proses penentuan jumlah klaster optimal, penelitian ini juga menghitung *Silhouette Score* sebagai metrik pendukung. *Silhouette Score* mengukur seberapa baik setiap titik data dikelompokkan ke dalam klasternya sendiri dibandingkan dengan klaster terdekat lainnya. Nilai ini dihitung untuk setiap titik data *i* menggunakan rumus:

$$s(i) = \frac{b(i) - a(i)}{\max\big(a(i), b(i)\big)} \tag{2.5}$$

di mana $a(i)$ adalah rata-rata jarak titik *i* terhadap seluruh titik lain dalam klaster yang sama (*intra-cluster distance*), dan $b(i)$ adalah rata-rata jarak titik *i* terhadap seluruh titik di klaster terdekat yang berbeda (*nearest-cluster distance*). *Silhouette Score* keseluruhan diperoleh sebagai rata-rata nilai $s(i)$ atas seluruh titik data. Nilai *Silhouette Score* berkisar antara -1 hingga $+1$, di mana nilai yang mendekati $+1$ menandakan bahwa titik data berada jauh lebih dekat ke klasternya sendiri dibandingkan ke klaster lain, sehingga mencerminkan kualitas pengelompokan yang baik.

2.3 FeedForward Neural Network (FNN)

Feedforward Neural Network (FNN) atau yang sering disebut sebagai *Multilayer Perceptron* (MLP) merupakan salah satu arsitektur dari *Artificial Neural Network* yang banyak digunakan untuk pemodelan prediktif pada data numerik. FNN terdiri dari beberapa lapisan *neuron* yang tersusun secara berurutan, yaitu *input layer*, satu atau lebih *hidden layer*, dan *output layer* [22]. Pada arsitektur ini, aliran informasi bergerak secara satu arah dari lapisan *input* menuju lapisan *output* tanpa adanya siklus atau umpan balik antar *neuron* [23]. Setiap *neuron* pada suatu lapisan terhubung dengan *neuron* pada lapisan berikutnya melalui bobot (*weights*) dan *bias* yang menentukan pengaruh masing-masing variabel masukan

terhadap keluaran jaringan. Struktur ini memungkinkan FNN untuk mempelajari hubungan non-linear antara variabel *input* dan *output* melalui transformasi berlapis menggunakan fungsi aktivasi non-linear. Arsitektur *multilayer perceptron* (MLP) yang merupakan bentuk dasar dari FNN telah terbukti mampu menghasilkan prediksi yang akurat pada data *time series* finansial, bahkan mengungguli model-model yang lebih kompleks seperti *recurrent neural network* dalam kondisi data tertentu [24]. Selain itu, penelitian empiris dalam konteks evaluasi risiko keuangan menunjukkan bahwa FNN secara konsisten menghasilkan performa prediksi yang kompetitif dibandingkan metode *machine learning* lainnya seperti *Support Vector Machine* dan *ensemble methods* [23].

Dalam penelitian ini, FNN berfungsi sebagai model prediksi yang dilatih secara terpisah untuk setiap klaster hasil *K-Means*. Model menerima fitur demografis dan finansial pengguna sebagai input dan menghasilkan prediksi alokasi pengeluaran pada 12 kategori sebagai output. Proses pencarian arsitektur terbaik untuk setiap model FNN dilakukan menggunakan *Grid Search*, sehingga konfigurasi *hyperparameter* yang digunakan pada masing-masing klaster merupakan konfigurasi yang secara sistematis menghasilkan performa prediksi alokasi keuangan yang paling optimal untuk segmen tersebut.

Secara matematis, proses perhitungan pada suatu *neuron* dalam jaringan dapat dinyatakan sebagai kombinasi linear antara *input* dan bobot yang kemudian ditransformasikan menggunakan fungsi aktivasi. Jika $x = (x_1, x_2, ..., x_n)$ merupakan vektor *input* dan $w = (w_1, w_2, ..., w_n)$ merupakan bobot yang terkait dengan *neuron* tersebut, maka nilai aktivasi awal (*weighted sum*) dapat dirumuskan sebagai:

$$z = \sum_{i=1}^{n} w_i x_i + b \tag{2.6}$$

di mana b merupakan *bias* dari *neuron*. Nilai tersebut kemudian diproses melalui fungsi aktivasi $f()$ untuk menghasilkan keluaran *neuron* yang dinyatakan sebagai:

$$a = f(z) \tag{2.7}$$

Pada *hidden layer*, fungsi aktivasi yang digunakan adalah *Rectified Linear Unit* (ReLU) yang berperan dalam memperkenalkan sifat *non-linear* pada model sehingga jaringan mampu menangkap pola hubungan kompleks dalam data. Secara matematis, fungsi ReLU didefinisikan sebagai:

$$f(z) = \max(0, z) \tag{2.8}$$

Fungsi ReLU menghasilkan nilai 0 untuk semua input negatif, dan mengembalikan nilai input itu sendiri untuk semua nilai positif. Sifat ini memungkinkan jaringan untuk tidak mengaktifkan seluruh neuron secara bersamaan sehingga komputasi menjadi lebih efisien dan masalah *vanishing gradient* dapat diminimalkan.

Pada *output layer*, fungsi aktivasi yang digunakan adalah fungsi *Sigmoid*, yang dipilih karena nilai target dalam penelitian ini berupa proporsi alokasi keuangan yang telah dinormalisasi ke dalam rentang $[0,1]$ menggunakan *MinMaxScaler*. Secara matematis, fungsi *Sigmoid* didefinisikan sebagai:

$$f(z) = \frac{1}{1+e^{-z}} \tag{2.9}$$

Fungsi *Sigmoid* secara alami membatasi keluaran *neuron* pada rentang $(0,1)$, sehingga sesuai untuk memprediksi nilai proporsi yang telah diskalakan.

Untuk jaringan berlapis, representasi matematis pada setiap lapisan l dapat diformulasikan sebagai berikut:

$$h^{(l)} = f\left(W^{(l)}h^{(l-1)} + b^{(l)}\right) \tag{2.10}$$

di mana $h^{(l)}$ adalah keluaran lapisan ke-l, $W^{(l)}$ adalah matriks bobot lapisan ke-l, $h^{(l-1)}$ adalah keluaran dari lapisan sebelumnya (atau vektor *input* jika $l = 1$), dan $b^{(l)}$ adalah vektor *bias* pada lapisan ke-l. Persamaan ini berlaku untuk setiap lapisan tersembunyi dalam arsitektur FNN yang digunakan.

Setelah setiap *hidden layer*, diterapkan *Batch Normalization* untuk menstabilkan distribusi aktivasi selama proses pelatihan. Teknik ini bekerja dengan menormalisasi keluaran suatu lapisan berdasarkan statistik *mini-batch* yang sedang diproses, sehingga mengurangi ketergantungan pelatihan terhadap inisialisasi bobot dan memungkinkan penggunaan *learning rate* yang lebih tinggi. Secara matematis, normalisasi terhadap suatu nilai aktivasi x_i dalam *mini-batch* $\mathscr{B}$ dilakukan sebagai:

$$\hat{x}_i = \frac{x_i - \mu_{\mathscr{B}}}{\sqrt{\sigma^2_{\mathscr{B}} + \varepsilon}} \tag{2.11}$$

di mana $\mu_{\mathscr{B}}$ adalah rata-rata dan $\sigma^2_{\mathscr{B}}$ adalah variansi dari *mini-batch*, serta ε adalah konstanta kecil untuk menjaga stabilitas numerik. Nilai yang telah dinormalisasi kemudian ditransformasikan kembali menggunakan parameter yang dapat dipelajari γ (*scale*) dan β (*shift*):

$$y_i = \gamma \hat{x}_i + \beta \tag{2.12}$$

Parameter γ dan β dipelajari selama proses pelatihan bersama dengan bobot jaringan lainnya, sehingga model tetap memiliki kapasitas untuk merepresentasikan distribusi aktivasi yang optimal.

Untuk mencegah *overfitting*, diterapkan *Dropout* setelah setiap lapisan *Batch Normalization*. *Dropout* bekerja dengan secara acak menonaktifkan sejumlah *neuron* selama proses pelatihan berdasarkan probabilitas p yang disebut *dropout rate*. Secara matematis, keluaran lapisan dengan *Dropout* dapat dirumuskan sebagai:

$$\tilde{\mathbf{h}}^{(l)} = \mathbf{h}^{(l)} \odot \mathbf{m}^{(l)}, \quad m_i^{(l)} \sim \text{Bernoulli}(1-p) \tag{2.13}$$

di mana $\odot$ menyatakan perkalian elemen per elemen (*element-wise*), $\mathbf{m}^{(l)}$ adalah vektor *mask* biner yang dibangkitkan dari distribusi Bernoulli dengan probabilitas keberhasilan $(1-p)$, dan p adalah *dropout rate*. Neuron yang dinonaktifkan (nilai *mask* sama dengan 0) tidak berkontribusi pada *forward pass* maupun *backpropagation* pada iterasi tersebut. Pada saat inferensi, *Dropout* dinonaktifkan dan seluruh *neuron* digunakan, sehingga keluaran jaringan bersifat deterministik.

Proses pembelajaran pada FNN dilakukan melalui mekanisme *forward propagation* dan *backpropagation*. Pada tahap *forward propagation*, data masukan diproses secara berlapis hingga menghasilkan prediksi keluaran $\hat{y}$. Oleh karena itu, model ini disebut *Feedforward Neural Network* (FNN) karena aliran informasi pada jaringan bergerak maju dari satu lapisan ke lapisan berikutnya [25]. Selanjutnya, kesalahan prediksi dihitung menggunakan fungsi kerugian (*loss function*) yang mengukur perbedaan antara nilai aktual y dan nilai prediksi $\hat{y}$. *Loss function* yang digunakan dalam penelitian ini adalah *Huber Loss*, dengan rumus matematis sebagai:

$$L_\delta(y,\hat{y}) = \begin{cases} \frac{1}{2}(y-\hat{y})^2, & \text{jika } |y-\hat{y}| \le \delta \\ \delta\left(|y-\hat{y}| - \frac{1}{2}\delta\right), & \text{jika } |y-\hat{y}| > \delta \end{cases} \tag{2.14}$$

Nilai kesalahan ini kemudian digunakan dalam proses *backpropagation* untuk memperbarui bobot jaringan. Proses *backpropagation* bekerja berdasarkan aturan

rantai (*chain rule*) kalkulus untuk menghitung gradien fungsi kerugian terhadap setiap bobot dalam jaringan. Secara matematis, gradien terhadap bobot w_i dihitung sebagai:

$$\frac{\partial L}{\partial w_i} = \left(\frac{\partial L}{\partial \hat{y}}\right) \cdot \left(\frac{\partial \hat{y}}{\partial z}\right) \cdot \left(\frac{\partial z}{\partial w_i}\right) \tag{2.15}$$

Berdasarkan nilai gradien yang diperoleh dari *backpropagation*, bobot jaringan diperbarui menggunakan metode optimisasi *Adaptive Moment Estimation* (Adam). Berbeda dengan *gradient descent* konvensional yang menggunakan *learning rate* tunggal untuk semua parameter, Adam mengadaptasi *learning rate* secara individual untuk setiap parameter dengan memanfaatkan estimasi momen pertama (rata-rata gradien) dan momen kedua (variansi gradien). Pada setiap langkah t, Adam memperbarui kedua estimasi momen tersebut sebagai berikut:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{2.16}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{2.17}$$

di mana $g_t = \frac{\partial L}{\partial w}$ adalah gradien pada langkah t, m_t adalah estimasi momen pertama (*mean* gradien), v_t adalah estimasi momen kedua (*uncentered variance* gradien), serta β_1 dan β_2 adalah koefisien peluruhan (*decay rates*) untuk masing-masing momen. Karena m_t dan v_t diinisialisasi dengan nol, keduanya mengalami bias ke arah nol pada iterasi awal. Untuk mengkoreksi bias ini, dilakukan koreksi bias sebagai berikut:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{2.18}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{2.19}$$

Selanjutnya, bobot jaringan diperbarui menggunakan estimasi yang telah dikoreksi:

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon} \hat{m}_t \tag{2.20}$$

di mana η adalah *learning rate* global, dan ε adalah konstanta kecil untuk menghindari pembagian dengan nol. Dengan mekanisme adaptif ini, Adam mampu menyesuaikan besaran pembaruan bobot secara otomatis berdasarkan riwayat gradien tiap parameter, sehingga proses konvergensi menjadi lebih stabil dan efisien dibandingkan *gradient descent* konvensional. Proses pembaruan bobot dilakukan secara iteratif hingga model mencapai konvergensi atau memenuhi kriteria penghentian pelatihan, sehingga jaringan mampu mempelajari pola hubungan antara variabel *input* dan target secara optimal.

2.4 Model Hybrid

Model *hybrid* dalam pembelajaran mesin merujuk pada pendekatan yang mengintegrasikan dua atau lebih metode dengan karakteristik berbeda untuk meningkatkan kinerja model secara keseluruhan. Dalam konteks sistem prediktif yang bekerja dengan data heterogen, pendekatan ini digunakan untuk menggabungkan keunggulan metode *unsupervised learning* dalam menemukan struktur tersembunyi dalam data dengan kemampuan *supervised learning* dalam melakukan pemodelan prediktif. Secara konseptual, model *hybrid* dibangun berdasarkan prinsip bahwa segmentasi data terlebih dahulu dapat mengurangi kompleksitas distribusi global dan menghasilkan subpopulasi yang lebih homogen, sehingga model prediksi pada masing-masing segmen menjadi lebih akurat dibandingkan model tunggal pada seluruh populasi. Hal ini didukung oleh studi terdahulu yang membuktikan bahwa penerapan *clustering* sebelum evaluasi secara nyata meningkatkan performa prediksi dari masing-masing model yang digunakan [26].

Secara spesifik dalam konteks yang serupa dengan penelitian ini, Chenghu dan Thammano [16] membuktikan bahwa kombinasi *K-Means* dan *neural network* secara *hybrid* menghasilkan peningkatan performa klasifikasi yang signifikan, di mana segmentasi awal menggunakan *K-Means* membantu *neural network* untuk lebih mudah mempelajari pola dalam subpopulasi yang lebih homogen. Temuan serupa juga diperoleh dalam konteks peringatan dini risiko keuangan, di mana penggabungan *K-Means* untuk kategorisasi kondisi finansial dengan *backpropagation neural network* untuk prediksi menghasilkan model yang mampu mengidentifikasi status keuangan secara akurat dengan tingkat kesalahan klasifikasi yang rendah [9, 27]. Dalam penelitian ini, model *hybrid* diimplementasikan melalui

tahap segmentasi menggunakan *K-Means* untuk membentuk kelompok pengguna dengan pola karakteristik serupa, kemudian dilanjutkan dengan pembangunan model *FeedForward Neural Network* terpisah pada setiap klaster untuk mempelajari pola alokasi keuangan secara spesifik. Pembagian peran antara kedua algoritma ini bersifat komplementer, dimana *K-Means* menjalankan fungsi klasifikasi dengan mengelompokkan pengguna ke dalam segmen-segmen yang homogen berdasarkan kesamaan karakteristik, sementara FNN dengan menggunakan *Grid Search* menjalankan fungsi optimasi dan prediksi dengan menemukan model terbaik yang mampu mengalokasikan keuangan secara akurat sesuai pola keuangan tiap segmen. Dengan arsitektur *hybrid* ini, kompleksitas distribusi data global dapat direduksi melalui segmentasi, sehingga setiap model FNN hanya perlu mempelajari pola dari subpopulasi yang lebih homogen dan terdefinisi dengan baik.

2.5 Metrik Evaluasi

Metrik evaluasi model merupakan instrumen kuantitatif yang digunakan untuk mengukur sejauh mana model yang dikembangkan mampu menghasilkan prediksi yang akurat dan dapat diandalkan. Pemilihan metrik evaluasi yang tepat tidak dapat dipisahkan dari karakteristik tugas prediksi yang dihadapi. Pada penelitian ini, model FNN digunakan untuk memprediksi nilai numerik kontinu berupa proporsi atau jumlah alokasi keuangan pada setiap kategori pengeluaran, sehingga evaluasi yang relevan adalah evaluasi regresi. Ketiga metrik yang digunakan dalam penelitian ini, yaitu *Root Mean Squared Error* (RMSE), *Mean Absolute Error* (MAE), dan *Mean Absolute Percentage Error* (MAPE), dipilih untuk saling melengkapi karena masing-masing mengukur aspek kesalahan prediksi dari perspektif yang berbeda. Kombinasi ketiga metrik ini memberikan gambaran evaluasi yang lebih komprehensif dibandingkan penggunaan satu metrik tunggal. Hal ini sejalan dengan kajian komparatif terhadap berbagai metrik evaluasi regresi yang menunjukkan bahwa penggunaan RMSE, MAE, dan MAPE secara bersamaan memberikan perspektif evaluasi yang lebih lengkap, karena masing-masing metrik mengukur karakteristik kesalahan prediksi dari sudut pandang yang berbeda dan saling melengkapi [28, 29].

2.5.1 Root Mean Square Error (RMSE)

Root Mean Square Error (RMSE) merupakan metrik evaluasi yang umum digunakan dalam masalah regresi untuk mengukur rata-rata besar kesalahan prediksi model terhadap nilai aktual. RMSE mengkuadratkan selisih antara nilai prediksi dan nilai aktual sehingga memberikan penalti yang lebih besar terhadap kesalahan dengan deviasi tinggi. Hal ini menjadikan RMSE sensitif terhadap *outlier* dan cocok digunakan ketika akurasi prediksi numerik menjadi prioritas utama.

Secara matematis, RMSE dirumuskan sebagai:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.21}$$

di mana:

- y_i adalah nilai aktual,
- $\hat{y}_i$ adalah nilai hasil prediksi model,
- n adalah jumlah data observasi [30].

Semakin kecil nilai RMSE, semakin tinggi tingkat akurasinya dan semakin baik kemampuan model dalam meminimalkan kesalahan prediksi [30, 31]. Dalam konteks penelitian ini, RMSE digunakan untuk mengevaluasi performa FNN dalam memprediksi proporsi atau jumlah alokasi anggaran pada masing-masing kalster, sehingga dapat diketahui tingkat akurasi model dalam menghasilkan prediksi finansial yang presisi.

2.5.2 Mean Absolute Error (MAE)

Mean Absolute Error (MAE) merupakan metrik evaluasi yang digunakan untuk mengukur rata-rata kesalahan absolut antara nilai prediksi dan nilai aktual [17]. Berbeda dengan RMSE, MAE tidak mengkuadratkan selisih kesalahan sehingga memberikan bobot yang sama pada setiap kesalahan tanpa memperbesar dampak dari *outlier*. Hal ini menjadikan MAE lebih *robust* terhadap data yang mengandung nilai ekstrem.

Secara matematis, MAE dirumuskan sebagai:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.22}$$

di mana:

- y_i adalah nilai aktual,
- $\hat{y}_i$ adalah nilai hasil prediksi model,
- n adalah jumlah data observasi [30].

Semakin kecil nilai MAE, semakin baik kemampuan model dalam melakukan prediksi. Nilai MAE berada dalam satuan yang sama dengan variabel target sehingga lebih mudah diinterpretasikan secara langsung. Dalam konteks penelitian ini, MAE digunakan sebagai metrik tambahan untuk mengevaluasi performa FNN dalam memprediksi proporsi atau jumlah alokasi anggaran pada masing-masing klaster, sehingga dapat diketahui rata-rata deviasi absolut antara nilai prediksi dan nilai aktual yang dihasilkan model.

2.5.3 Mean Absolute Percentage Error (MAPE)

Mean Absolute Percentage Error (MAPE) merupakan metrik evaluasi yang mengukur rata-rata persentase kesalahan absolut antara nilai prediksi dan nilai aktual. MAPE menyatakan kesalahan dalam bentuk persentase sehingga bersifat *unit-free* atau tidak bergantung pada satuan data, yang menjadikannya mudah untuk diinterpretasikan dan dibandingkan antar model maupun antar dataset yang berbeda skala.

Secara matematis, MAPE dirumuskan sebagai:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100\% \tag{2.23}$$

di mana:

- y_i adalah nilai aktual,
- $\hat{y}_i$ adalah nilai hasil prediksi model,
- n adalah jumlah data observasi [30].

Nilai MAPE yang lebih rendah menunjukkan tingkat akurasi model yang lebih tinggi. Secara umum, model dianggap memiliki akurasi yang sangat baik apabila nilai MAPE berada di bawah 10% [32], akurasi baik pada kisaran 10%–20%, dan kurang akurat apabila melebihi 50% [28]. Namun perlu

diperhatikan bahwa MAPE dapat menghasilkan nilai yang tidak terdefinisi atau sangat besar apabila terdapat nilai aktual yang mendekati atau sama dengan nol. Dalam konteks penelitian ini, MAPE digunakan untuk mengevaluasi performa FNN dalam menghasilkan prediksi alokasi anggaran yang presisi, dengan mengukur seberapa besar persentase deviasi prediksi model terhadap nilai aktual pada setiap klaster.

2.6 Tabel Penelitian Terkait

Tabel 2.1 menyajikan tinjauan terhadap berbagai penelitian terdahulu yang relevan dengan topik penelitian yang dilakukan. Tinjauan ini bertujuan untuk memberikan gambaran mengenai perkembangan penelitian sebelumnya, khususnya terkait metode yang digunakan serta hasil yang diperoleh.

Tabel 2.1. Penelitian Terdahulu

No	Judul	Penulis	Algo/Metode	Hasil
1.	Design of a Rule-based Personal Finance Management System based on Financial Well-being	Alhanoof Althnian (2021)	Rule-Based System	Sistem yang memberikan saran tentang pengeluaran dan tabungan yang memungkinkan pengguna untuk membuat keputusan pengeluaran yang terinformasi dan mencapai tujuan keuangan mereka.
2.	Developing a Rule-Based System to Recommend Household Budget	Sonia Dey (2025)	Rule-Based System, KNN, Naive Bayes	KNN mencapai akurasi tertinggi sebesar 96,67%, sementara Naive Bayes mencapai akurasi 93,00%, dan Rule-Based System mencapai akurasi 90,00%.
Lanjut ke halaman berikutnya...				

Tabel 2.1 Lanjutan

No	Judul	Penulis	Algo/Metode	Hasil
3.	AI-Based Wealth Advisory System using Machine Learning and Predictive Analytics for Personalized Budget Planning	Bhavana B R (2025)	Arsitektur multi-model yang dirancang untuk klasifikasi, prediksi, deteksi anomali, dan rekomendasi yang dipersonalisasi.	Akurasi klasifikasi pengeluaran = 91% F1-score; Error prediksi = MAE $43/bulan per pengguna; Akurasi deteksi anomali = 95%; Rekomendasi adoption rate = 41%.
4.	Application of Cluster Analysis in Optimizing Financial Product Recommendation System	Ni Rui (2025)	K-Means Clustering; DBSCAN; Hierarchical Clustering	Clustering terbukti meningkatkan matching antara user dan produk; meningkatkan personalization, hasilnya adalah rekomendasi produk keuangan berbasis cluster bukan alokasi budget numerik.
5.	Personal Finance Manager with Predictive Analytics Using LSTM for Enhanced Financial Decision Making	Jatin Kumar (2025)	Long Short-Term Memory (LSTM)	Model yang mampu memprediksi tren keuangan pengguna dengan akurasi sekitar 85%.

Lanjut ke halaman berikutnya...

Tabel 2.1 Lanjutan

No	Judul	Penulis	Algo/Metode	Hasil
6.	Predictive Analytics for Personalized Debt Management: Leveraging Machine Learning to Provide Actionable Financial Advice	Amir Ahmad Dar (2025)	Logistic Regression; Random Forest; K-Means; KNN; DBSCAN; Rule-based system	kategori risiko peminjam (low, medium, high risk) dan rekomendasi pengelolaan utang yang disesuaikan dengan profil risiko.
7.	SmartSave: A Machine Learning-Driven Financial Analysis Platform	Dhanunjay J (2025)	K-Means Clustering, Rule-Based Budget, Recommendation (50/30/20 rule dan pay-yourself-first)	Sistem yang berhasil mengelompokkan pengguna ke dalam beberapa profil keuangan yang berbeda dan memberikan rekomendasi budgeting meskipun masih menggunakan Rule-Based System.
Lanjut ke halaman berikutnya...				

Tabel 2.1 Lanjutan

No	Judul	Penulis	Algo/Metode	Hasil
8.	WONGA: The Future of Personal Finance Management – A Machine Learning-Driven Approach for Predictive Analysis and Efficient Expense Tracking	Uyanahewa M.I.R (2023)	NLP, ANN, CNN, Multinomial Naive Bayes, ARIMA, LSTM	Sebuah aplikasi yang mampu melacak arus kas melalui pesan SMS bank dan tagihan belanja, mengategorikan pengeluaran secara otomatis, memprediksi pengeluaran masa depan berdasarkan kalender digital, serta membuat rencana anggaran khusus.
9.	The Model of Food Nutrition Feature Modeling and Personalized Diet Recommendation Based on the Integration of Neural Networks and K-Means Clustering	Pei-Min Lu (2025)	K-Means Clustering, Artificial Neural Network (ANN), Decision Tree, Gradient Boosted Decision Tree, dan K-Nearest Neighbors	Sistem berhasil mengelompokkan makanan berdasarkan karakteristik nutrisi dan memetakan kebutuhan nutrisi pengguna ke cluster yang sesuai, sehingga rekomendasi yang dihasilkan selaras dengan tujuan diet pengguna.

Lanjut ke halaman berikutnya...

Tabel 2.1 Lanjutan

No	Judul	Penulis	Algo/Metode	Hasil
10.	Study of an Adaptive Financial Recommendation Algorithm Using Big Data Analysis and User Interest Pattern with Fuzzy K-Means Algorithm	Jinyong Yang (2024)	Fuzzy K-Means Clustering, Neural Collaborative Filtering (NCF)	algoritma FNFinRec (Fuzzy Neural Financial Recommendation) yang dirancang untuk memberikan rekomendasi keuangan yang adaptif dan akurat.