Vol.7 No.1 Tahun 2026
Jurnal Kecerdasan Buatan, Komputasi dan Teknologi Informasi
https://ejournal.unuja.ac.id/index.php/core
E-ISSN : 2774-7875 and P-ISSN : 2775-0124

Implementation of Artificial Neural Networks in an Android-Based Online Course Recommendation Application

Muhamad Syaiful Anwar[1], Murniyati[2]
[1,2] Gunadarma University, Depok, Indonesia

Article Info

Article history:

Received : mm dd, yyyy
Revised : mm dd, yyyy
Accepted : mm dd, yyyy

Keywords:

Neural Network;
Recommendation;
Course;
Online;
Android;

Abstract

The growing interest in online courses has caused information overload. Users now struggle to choose suitable programs. Machine Learning (ML) can address this problem through a Recommendation System Model based on an artificial neural network (ANN). This research aims to address this issue by designing an intelligent, ML-based recommendation system integrated within an Android app. The system uses a TensorFlow-built ML model to provide personalized course recommendations. The native Android application is developed with Kotlin and Jetpack Compose. Results show that the developed model provides recommendations that match user preferences and is successfully implemented in the online course recommendation application 'Rekomendasi Kursus Online.' Supporting features include Google account authentication, cloud storage, synchronization for favorite course lists, and dark mode. These features work as expected. User testing with the System Usability Scale (SUS) yielded a score of 83.17. This places the app in the Grade B category with a "Good" rating, showing the application is well-designed and suitable for use.

Corresponding Author:

Muhamad Syaiful Anwar
syaiful.sa95@gmail.com

INTRODUCTION

Education serves as the primary foundation for a nation's progress and a principal medium for developing individual potential. In the global dynamic, the digital era has driven fundamental transformations across various sectors, including the education system in Indonesia. Alongside technological advancements, there has been a significant increase in public interest in online courses as a means for skill development and career exploration. E-learning platforms have evolved into alternative educational ecosystems that offer flexibility and accessibility without geographical boundaries. This phenomenon, while positive, introduces a new challenge known as information overload. Prospective learners are confronted with a massive volume of course options, often experiencing difficulty in identifying programs that best align with their interests, professional goals, and competency levels. This condition potentially leads to confusion, decision-making procrastination, and suboptimal choices (Hidayatullah et al., 2021).

To overcome the complexity of course selection, Machine Learning (ML) based recommendation systems offer an effective solution. A recommendation system is defined as a technology capable of filtering massive data volumes to present the most relevant courses for each user personally (Ricci et al., 2022). By applying Machine Learning algorithms, the

system can learn user preferences based on historical interaction data to generate dynamic and accurate recommendations.

In recent years, various studies have highlighted a shift in recommendation system research from conventional methods to the utilization of modern techniques (Ko et al., 2022). Furthermore, systematic evaluations indicate that the integration of Deep Learning and hybrid approaches has become essential in managing the increasing data complexity in e-learning (Roy & Dutta, 2022). Specifically, the utilization of Artificial Neural Network architectures has proven highly effective in handling intricate nonlinear relationships among user interaction variables (Houssein et al., 2022). Additionally, the deployment of such deep learning-based models has been acknowledged to significantly improve prediction accuracy compared to traditional algorithms (Zheng et al., 2023). Nevertheless, deep learning models generally require substantial computational power, often limiting their implementation to server or cloud environments. A distinct research gap exists regarding the optimization of neural network models for efficient, localized execution on mobile devices (edge AI). In particular, there is a scarcity of literature addressing the end-to-end integration of optimized recommendation models into Android-based applications with modern, responsive interfaces.

Therefore, this research focuses on the design, development, and implementation of a Machine Learning model for an online course recommendation system. The predictive model is constructed using the TensorFlow framework, adopting a neural network-based approach to fulfill the system's accuracy requirements. Subsequently, the model is optimized using TensorFlow Lite to enable local and efficient execution on mobile devices(Siregar & Wijayanti, 2025). The final solution is implemented in an Android-based application utilizing Jetpack Compose to build a modern and responsive user interface. Furthermore, the Firebase platform is integrated to support additional functionalities, thereby enhancing the user experience. Through this approach, the research aims to produce a utility instrument that is not only technically superior but also accessible and capable of providing practical benefits to the community(Khodijah et al., 2024).

This study specifically aims to build a neural network-based Machine Learning model using TensorFlow for an online course recommendation system, optimize the model using TensorFlow Lite for implementation within an Android online course recommendation application, and integrate several supporting features utilizing Firebase to elevate the user experience within the Android application(Cania & Jehwae, 2025).

This research employs the Waterfall software development methodology, a sequential model consisting of well-defined phases that must be completed in order. The methodological workflow encompasses Planning, Analysis, Design, Implementation, and Testing(Nuryansyah & Ratnawati, 2020).

METHOD

This study employs the Waterfall software development method, a sequential model consisting of well-defined phases that must be completed in order. This method is one of the most traditional and sequential software development life cycle (SDLC) models(Chandani et al., 2022). Its main characteristic is its linear and well-defined structure, where documentation becomes an essential output of each phase. The Waterfall method is suitable for projects where system requirements are well understood from the beginning and are not anticipated to undergo many changes during the development process (Saputro et al., 2020). The methodological workflow encompasses Planning, Analysis, Design, Implementation, and Testing.

DOI: https://doi.org/10.33650/coreai.v6i2

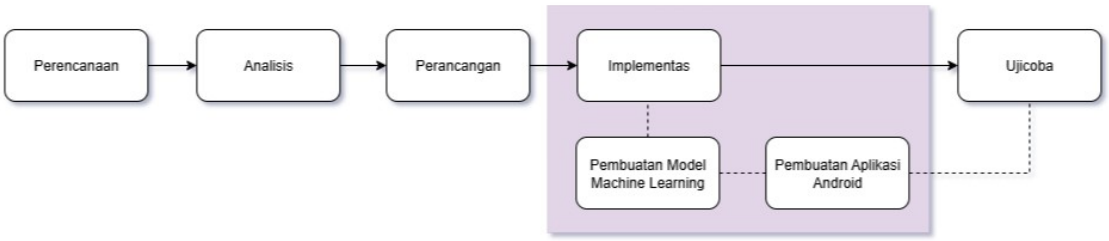

Figure. 1. Study Method

1. Planning and Analysis

The application is designed to help users quickly find suitable courses through preference-based recommendations. The minimum viable functions include: preference input (topic/sub-category, course type, duration), displaying accessible recommendations, saving favorites, Google authentication, favorites synchronization, and dark mode(Nopriandi & Haswan, 2022). An on-device inference approach was chosen to reduce latency and allow limited offline usage. The analysis phase was conducted to define the system requirements in detail. Functional analysis established mandatory features such as user authentication, the recommendation system, favorite storage with cloud synchronization, and dark mode. Non-functional analysis documented technical requirements, including hardware (a laptop with a minimum specification of an AMD Ryzen 3 and 16 GB of RAM) and software (Android Studio, Visual Studio Code).

2. System Design

This recommendation system is designed using the Content-Based Filtering (CBF) method. This approach was selected due to its ability to provide personalized recommendations based on user-provided preferences, without requiring historical data from other users (Al-Ghuribi & Noah, 2021). This is highly effective in overcoming the cold-start problem for new users. The system flow begins when the user inputs preferences, which then serve as input for the ML model to predict the most relevant course categories.

Meanwhile, the software architecture of this Android application utilizes the Model-View-ViewModel (MVVM) pattern. This pattern strictly separates the user interface (View) components from the business and data logic (Model) (Wibowo, 2021). The ViewModel acts as a bridge, exposing relevant data from the Model to the View and handling user interactions. The main advantages of this architecture are enhanced modularity, testability, and code readability. The Firebase platform is also integrated as a Backend-as-a-Service (BaaS). This platform provides a suite of ready-to-use services that handle various server-side requirements, allowing developers to focus more on client-side interface development (Yaseen et al., 2022).

Furthermore, to ensure the application is user-friendly, the user flow and wireframes were designed in detail. The user flow maps the steps users will take to achieve various goals within the application, ranging from the authentication process and filling out the preference form to viewing recommendation results, saving favorites, and accessing settings. Wireframes provide a visual representation of each screen's layout to guide the interface development process.

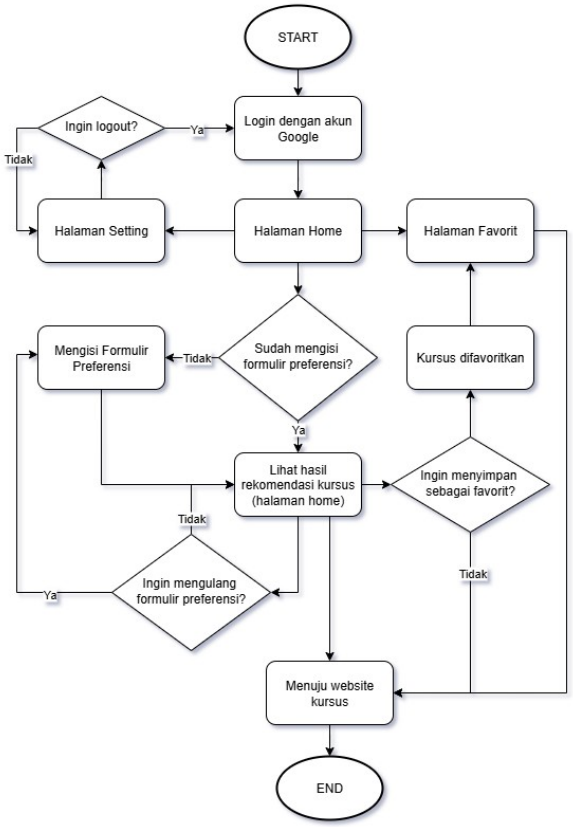

Figure. 2. Userflow

3. Implementation and Testing

The implementation phase includes creating the Machine Learning model and developing the Android application. The testing phase was conducted to validate functionality and measure system usability. Functionality testing was performed using the Black Box Testing method. This approach focuses entirely on testing functionality from the user's perspective by providing inputs and evaluating the outputs generated by the system (Sib, 2021). Subsequently, a usability evaluation was conducted using the System Usability Scale (SUS) method to measure the application's feasibility by users.

RESULTS AND ANALYSIS

1. Implementation and Results of the Machine Learning Model

Model development was conducted in a Jupyter Notebook environment using Python. This phase began with dataset preparation and data pre-processing. The raw Coursera course dataset from Kaggle, comprising 8,092 rows and 45 columns, served as the starting point. The data cleaning process initiated with the selection of the 9 most relevant features and the elimination of rows containing missing values, resulting in a clean dataset of 2,775 rows. To prepare the data for the model, categorical features (Sub-Category, Course Type, Category) were transformed into numerical representations using LabelEncoder. Meanwhile, the numerical feature, Duration, was normalized using StandardScaler to ensure a uniform value scale. Subsequently, the dataset was divided into two parts, where 80 percent was used for training data and 20 percent was used for testing data. The Artificial Neural Network model was then constructed utilizing the Keras API from TensorFlow. The model architecture is sequential, consisting of one input layer, five hidden layers with ReLU activation functions, and one output layer with a Softmax activation function for multiclass classification. During training, an Early Stopping mechanism was employed to prevent overfitting.

DOI: https://doi.org/10.33650/coreai.v6i2

```
17/17 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 0.9297 - loss: 0.3003
Model Accuracy: 93.4%
```

Figure. 3. Model's Accoracy Result

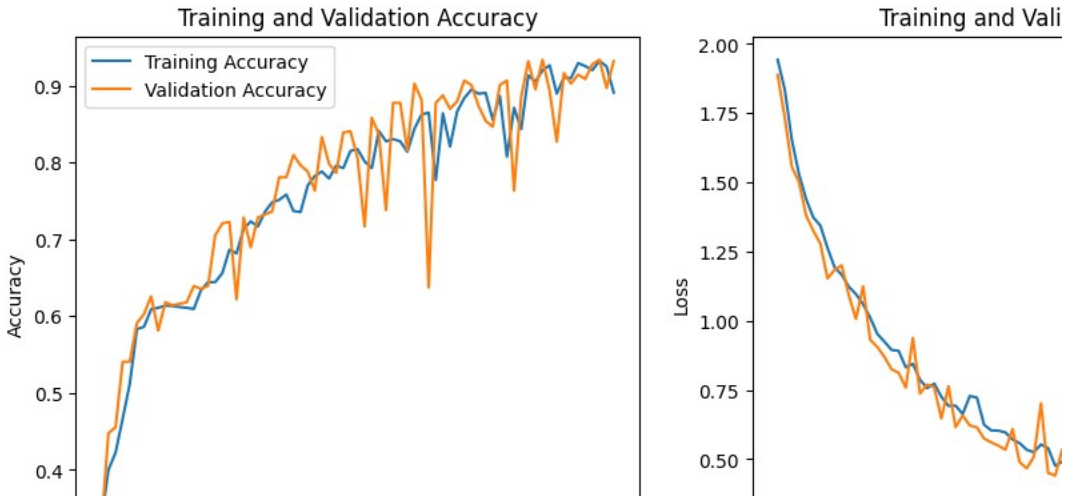

Figure. 4. Model's Learning Graph

The results of this model training indicate that the model achieved an accuracy rate of 93.4% on the testing data. Meanwhile, the training curve analysis demonstrates good convergence, where validation accuracy increases alongside training accuracy, and loss decreases consistently. To implement the model into the Android application, it was converted into the TFLite format.

2. Implementation and Testing of the Android Application

The Android application was developed natively using Kotlin and Jetpack Compose, adhering to the MVVM architecture. The first step taken was to integrate the model and the recommendation logic. The file was integrated into the application's assets. Subsequently, a wrapper class (TFLiteHelper.kt) was created to handle the entire inference logic. This class loads the model, pre-processes the input from the user form to match the format expected by the model, executes interpreter.run(), and post-processes the output to obtain the predicted category.

Next, the Room Persistence Library was utilized to create a local SQLite database that stores course data and favorite statuses. Firebase was integrated as the Backend-as-a-Service for this application. Firebase Authentication handles the secure login process with Google, while Cloud Firestore is used to store the user's favorite course IDs, enabling real-time data synchronization across devices.

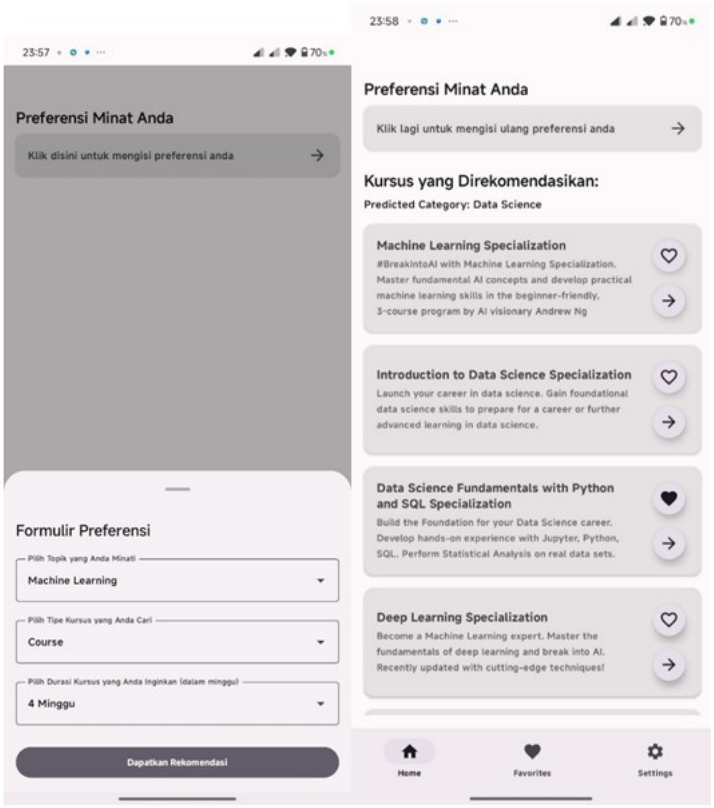

Figure. 5. Homepage and Preference Form Final Result

3. Functionality Testing Results and System Usability Scale

Functionality testing was conducted using the Black Box Testing method. Every testing scenario, ranging from login, form completion, and recommendation display, to favorite data synchronization after re-login, was evaluated. The testing results indicate that all implemented features function as expected and are declared valid.

Table 1. Black Box Testing Results on Android Application Functionality

No.	Tested Feature	User Interaction	Expected Output	Result
1	Firebase Authentication Login Feature	The user presses the login button and logs in with a Google Account	The user successfully logs in and is redirected to the main page	Valid.
2	Course Recommendation Feature	The user presses the button to fill out the preference form, fills in 3 inputs on the form, and presses the get recommendation button	The model will process the input, predictm and display the top 10 course recommendations on the home page	Valid.
3	Feature to save course recommendation as a favorite	The user presses the favorite button on one of the course recommendations	The favorited course recommendation will be saved in the favorites page	Valid.
4	Dark mode feature	The user presses the switch to activate the	The application's appearance	Valid.

DOI: https://doi.org/10.33650/coreai.v6i2

		application's dark mode	become dark, pressing the button again will revert it to light	
5	Firestore Database synchronization feature	The user logs in with Google account A and saves several course recommendations. Logs out by pressing the logout button on the settings page, logs in with Google account B, gets course recommendations, and saves several course recommendations. Logs out from Google account B, then logs back in with Google account A.	Favorited course recommendations will be saved automatically in the Firestore Database; when the user logs in with the same Google account, the saved course recommendations will reappear on the favorites page.	Valid.

Subsequently, a usability evaluation was conducted using the System Usability Scale (SUS) method involving 30 respondents. SUS is an industry-standard instrument used to measure users' subjective perceptions regarding the ease of use of a system. The analysis results from the questionnaire yielded a final SUS score of 83.17. Based on the standard grading scale, this score falls into the Grade B category with a 'Good' predicate, indicating that the application is well-received by users and considered easy to use.

CONCLUSIONS

Based on the conducted research, several conclusions can be drawn. First, the developed machine learning model is capable of providing course recommendations that align with user preferences. The model was successfully implemented into the Online Course Recommendation Android application and operates effectively. Furthermore, other supporting features, such as Google account authentication, cloud storage and synchronization for favorite course lists, and the application's dark mode, have been integrated and function seamlessly. Based on the user testing results of this Android application using the System Usability Scale, a final score of 83.17 was obtained. This score falls into Grade B with an adjective rating of "Good," indicating that the Android application design works well and is highly feasible for use.

Although this research has successfully achieved its objectives, there is still room for further development. Suggestions that the author can provide for consideration in future developments include expanding the model's dataset to recommend online courses other than Coursera. The model's implementation can also be enhanced by adopting a hybrid machine learning computation approach, namely offline (on-device) and online (cloud computing) executions. This can be done to accommodate users who lack sufficiently powerful computational capabilities on their devices, as well as to improve the computational scalability of the model itself. Additionally, the application's user interface (UI) can be optimized to be more streamlined and lightweight, making it easier to understand for various demographics of users.

REFERENCES

Cania, O., & Jehwae, A. (2025). Implementasi Perancangan Sistem Informasi Berbasis Website di SD Negeri 18 Angge Palimbatan. *Journal of Information System and Education Development*, *3*(4), 4–8. https://doi.org/10.62386/JISED.V3I4.152

Chandani, C., Migunani, M., & Putra, T. W. A. P. (2022). RANCANG BANGUN SISTEM INFORMASI AKADEMIK BERBASIS WEB MOBILE. *Jurnal Teknik Informatika Dan Teknologi Informasi*, *1*(3), 08–19. https://doi.org/10.55606/JUTITI.V1I3.62

Khodijah, K., Sriyanto, S., Aziz, R. Z. A., & Suhendro, S. (2024). PERBANDINGAN KINERJA LIMA ALGORITMA KLÀSIFIKASI DASAR UNTUK PREDIKSI PENYAKIT JANTUNG "CLASSIFIER: NB, DTC4. 5, KNN, ANN & SVM". *Jaringan Sistem Informasi Robotik-JSR*, *8*(2), 230–234.

Nopriandi, H., & Haswan, F. (2022). Analisis Klasterisasi Mahasiswa Baru dalam Memilih Program Studi dengan Menggunakan Algoritma K-Means. *Journal of Information System Research (JOSH)*, *3*(4), 666–671. https://doi.org/10.47065/JOSH.V3I4.1986

Nuryansyah, A., & Ratnawati, D. (2020). Pengembangan Sistem Informasi Sekolah Berbasis Website Di SMK Taman Karya Madya Ngemplak. *JINTECH: Journal Of Information Technology*, *1*(2), 21–31. https://doi.org/10.22373/JINTECH.V1I2.593

Siregar, J., & Wijayanti, D. (2025). Integrasi Model Neural Network dengan SVM dan KNN untuk Deteksi Dini Penyakita Jantung pada Penderita Hypertensi. *Technologia: Jurnal Ilmiah*, *16*(1), 104–109.

Al-Ghuribi, S. M., & Noah, S. A. M. (2021). A comprehensive overview of recommender system and sentiment analysis. arXiv:2109.08794.

Hidayatullah, F. A., Haviluddin, H., & Alfred, R. (2021). Systematic review of information overload on social media. Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), 5(2), 387-394.

Houssein, E. H., et al. (2022). Deep learning for recommender systems: A review. Neural Computing and Applications, 34, 21423-21442. https://doi.org/10.1007/s00521-022-07445-5

Ko, H., Lee, S., Park, Y., & Choi, A. (2022). A survey of recommendation systems: Recommendation models, techniques, and application fields. Electronics, 11(1), 141. https://doi.org/10.3390/electronics11010141

Ricci, F., Rokach, L., & Shapira, B. (2022). Recommender systems: Techniques, applications, and challenges. In Recommender systems handbook (pp. 1-42). Springer. https://doi.org/10.1007/978-1-0716-2197-4_1

Roy, D., & Dutta, M. (2022). A systematic review and research perspective on recommender systems. Journal of Big Data, 9(1), 59. https://doi.org/10.1186/s40537-022-00592-5

Saputro, N. H., Aripradono, A. D., & Baqi, M. P. A. (2020). Perancangan sistem informasi pendaftaran siswa baru berbasis website (Studi kasus: SMAN 12 Tangerang Selatan). Jurnal Informatika Universitas Pamulang, 5(4), 525-531.

Sib, M. (2021). Pengujian black box pada aplikasi perpustakaan menggunakan metode equivalence partitions. Jurnal Teknologi Sistem Informasi dan Aplikasi, 4(2), 94-99. https://doi.org/10.32493/jtsi.v4i1.8519

Wibowo, A. S. (2021). Perancangan aplikasi point of sales (POS) pada platform Android dengan arsitektur MVVM (Model-View-ViewModel). Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 5(7), 2807-2814.

Yaseen, Q. I., Al-Slamy, N. M., & Al-Tememi, A. A. (2022). A comprehensive review of Google Firebase in mobile and web applications. International Journal of Interactive Mobile Technologies (iJIM), 16(10), 4-22.

Cania, O., & Jehwae, A. (2025). Implementasi Perancangan Sistem Informasi Berbasis Website di SD Negeri 18 Angge Palimbatan. Journal of Information System and Education Development, 3(4), 4-8. https://doi.org/10.62386/JISED.V3I4.152

https://doi.org/10.62386/jised.v3i4.152

Chandani, C., Migunani, M., & Putra, T. W. A. P. (2022). RANCANG BANGUN SISTEM INFORMASI AKADEMIK BERBASIS WEB MOBILE. Jurnal Teknik Informatika Dan Teknologi Informasi, 1(3), 08-19. https://doi.org/10.55606/JUTITI.V1I3.62 https://doi.org/10.55606/jutiti.v1i3.62

Khodijah, K., Sriyanto, S., Aziz, R. Z. A., & Suhendro, S. (2024). PERBANDINGAN KINERJA LIMA ALGORITMA KLÀSIFIKASI DASAR UNTUK PREDIKSI PENYAKIT JANTUNG "CLASSIFIER: NB, DTC4. 5, KNN, ANN & SVM". Jaringan Sistem Informasi Robotik-JSR, 8(2), 230-234.

Nopriandi, H., & Haswan, F. (2022). Analisis Klasterisasi Mahasiswa Baru dalam Memilih Program Studi dengan Menggunakan Algoritma K-Means. Journal of Information System Research (JOSH), 3(4), 666-671. https://doi.org/10.47065/JOSH.V3I4.1986 https://doi.org/10.47065/josh.v3i4.1986

Nuryansyah, A., & Ratnawati, D. (2020). Pengembangan Sistem Informasi Sekolah Berbasis Website Di SMK Taman Karya Madya Ngemplak. JINTECH: Journal Of Information Technology, 1(2), 21-31. https://doi.org/10.22373/JINTECH.V1I2.593 https://doi.org/10.22373/jintech.v1i2.593

Siregar, J., & Wijayanti, D. (2025). Integrasi Model Neural Network dengan SVM dan KNN untuk Deteksi Dini Penyakita Jantung pada Penderita Hypertensi. Technologia: Jurnal Ilmiah, 16(1), 104-109. https://doi.org/10.31602/tji.v16i1.17046

Zheng, Y., et al. (2023). *A survey on deep learning for recommender systems*. arXiv.