Jurnal Kependidikan:
Jurnal Hasil Penelitian dan Kajian Kepustakaan di Bidang Pendidikan, Pengajaran dan Pembelajaran
https://e-journal.undikma.ac.id/index.php/jurnalkependidikan/index

Vol. 10, No. 4 : December 2024
E-ISSN: 2442-7667
pp. 1355-1366
Email: jklppm@undikma.ac.id

The Enterprise School Readiness Prediction System (ESRPS) Uses Machine Learning to Assess Children's Readiness for Entering Elementary School

Muhammad Choerul Umam[1*], Cicilia Dyah Sulistyaningrum I.[2], Dydik Kurniawan[3], Priyono Tri Febrianto[4]
[1*,2,]Department of Office Administration Education, Universitas Sebelas Maret, Indonesia.
[3]Department of Office Guidance and Counseling, Universitas Mulawarman, Indonesia.
[4]Department of Office Primary Education, Universitas Trunojoyo, Indonesia.
*Corresponding Author. Email: mc_umam07@staff.uns.ac.id

Abstract: This study aims to develop and evaluate the Enterprise School Readiness Prediction System (ESRPS) to predict children's readiness for elementary school using machine learning algorithms. This research employs the Research and Development (R&D) method using Borg and Gall's model and Instruments include questionnaires, programming tools, performance evaluation metrics, and web/database development tools to ensure the system's validity, reliability, and practical applicability.The research analyzes data from 300 students in various Indonesian cities, focusing on attributes like age, gender, and parental education. The system implements four algorithms: Decision Tree, Random Forest, Naive Bayes, and SVM. Data preprocessing, model training, and hyperparameter tuning were conducted, followed by evaluation using metrics like accuracy and precision. A web-based application was developed for user interaction and deployment. The result showed that the Decision Tree and Naive Bayes algorithms achieved the highest accuracy at 55%, followed by SVM at 50%, and Random Forest at 45%. This suggests that simpler models may be more suitable for the dataset's characteristics. The system also demonstrated the feasibility of practical deployment for educational use. The study concludes that ESRPS effectively uses machine learning to assess school readiness, highlighting the value of data preprocessing and model tuning in enhancing accuracy. Despite moderate accuracy levels, the study confirms the system's potential for aiding educators and parents in supporting children's transition to school.

Article History
Received: 09-10-2024
Revised: 14-11-2024
Accepted: 29-11-2024
Published: 21-12-2024

Key Words:
Machine Learning; Naive Bayes; Random Forest; Decision Trees; School Readiness.

How to Cite: Umam, M., Sulistyaningrum I., C., Kurniawan, D., & Febrianto, P. (2024). The Enterprise School Readiness Prediction System (ESRPS) Uses Machine Learning to Assess Children's Readiness for Entering Elementary School. *Jurnal Kependidikan: Jurnal Hasil Penelitian dan Kajian Kepustakaan di Bidang Pendidikan, Pengajaran dan Pembelajaran, 10*(4), 1355-1366. doi:https://doi.org/10.33394/jk.v10i4.13488

https://doi.org/10.33394/jk.v10i4.13488

Introduction

School readiness is a multifaceted concept that encompasses a child's physical, social, emotional, and cognitive development, all of which are essential for success in a school environment (Shaw et al., 2021). School readiness is a combination of factors that influence a child's ability to participate in formal educational activities. This includes academic, social, emotional, and communication skills (Gill et al., 2020). According to the Minister of Education Regulation Number 1 of 2021 Article 2, the implementation of the New Student Admission (PPDB) for first-grade elementary school prioritizes the acceptance of prospective students who are 7 years old or at least 6 years old for those with exceptional intelligence or talent and mental readiness. The age of seven is considered appropriate for children to enter elementary school because, at this age, children have developed the physical readiness and psychological readiness to participate in the elementary education process (Jannah, 2023). Conversely, being under the age of 7 is deemed less suitable because children are not yet capable of navigating the school system (Dockett & Perry, 2002). This unpreparedness can result in communication problems and emotional regulation challenges during adolescence and into adulthood (Pekdogan & Akgul, 2016), However, not every child has the same physical and psychological condition at a certain age. Gender can also influence the differences in children's developmental stages (Halmatov, 2018). Therefore, there is a need for a prediction system model that is suitable for assessing children's readiness for school. This model should take into account various factors, including physical and psychological conditions, as well as gender differences, to provide a comprehensive evaluation of each child's unique developmental stage. By utilizing such a prediction system, educators and parents can better understand the individual needs of children, ensuring that they receive the appropriate support and interventions to facilitate a smooth transition into the educational environment (Kokkalia et al., 2019).

Machine learning can serve as an alternative solution for developing this prediction system model (Lakkaraju et al., 2015). By leveraging algorithms and data analysis, machine learning can analyze various factors influencing a child's readiness for school, such as physical and psychological conditions, developmental milestones, and demographic variables like gender (Al Mayahi & Al-Bahri, 2020; Lakkaraju et al., 2015). Algorithms that have been used in predicting school readiness include Decision Tree, K-Nearest Neighbor (KNN), and Artificial Neural Network (ANN) (Cattell & Bruch, 2021). These algorithms are popular in educational data mining due to their ability to handle complex data and provide insights into various factors affecting children's readiness for school (Nurlina, 2019). K-Nearest Neighbor (KNN) is a powerful algorithm used for classification and regression tasks, identifying the 'k' nearest data points to make predictions (Nuranisah et al., 2020). In the context of school readiness, KNN can classify children based on their developmental attributes to identify those who may need additional support. Similarly, Artificial Neural Network (ANN) mimics the human brain's neural network, allowing it to recognize patterns and learn from data (Pregowska & Osial, 2021). ANN is particularly effective in complex predictions involving nonlinear relationships among input features and can assess various factors affecting a child's readiness for school. However, there are several research gaps in using KNN and ANN for predicting school readiness. First, there is a need for refined feature extraction techniques to improve model accuracy.

Additionally, hybrid models that combine the strengths of both algorithms could enhance performance across different dataset sizes (Cattell & Bruch, 2021). The applicability of existing models in diverse demographic settings also highlights the necessity for culturally sensitive approaches. Furthermore, the lack of longitudinal studies examining how children's readiness evolves over time indicates a need for temporal analysis. Lastly, incorporating diverse data sources, such as behavioral, cognitive, and social-emotional data, into predictions is essential for creating comprehensive models. Given the weaknesses of the aforementioned algorithms, this study will employ three other algorithms, namely Support Vector Machine (SVM), Naive Bayes, Random Forest, and Decision Trees, which are known to have higher accuracy in various classification applications, including in predicting school readiness. By leveraging the strengths of these algorithms, it is expected that the prediction results will be more accurate and effective. The research aims to develop and evaluate the Enterprise School Readiness Prediction System (ESRPS) using machine learning to predict children's readiness for elementary school based on demographic and developmental factors. It seeks to assist educators and parents in supporting children's transition to formal education through data-driven insights. The study's novelty lies in applying machine learning (Decision Tree, Naive Bayes, Random Forest, SVM) to school readiness assessment, demonstrating the effectiveness of simpler models, emphasizing data preprocessing and hyperparameter tuning, and deploying a user-friendly web-based application. It also uses a localized dataset, offering insights into regional educational contexts in Indonesia.

Research Method

This research uses the Research and Development (R&D) method to design, develop, and evaluate the Enterprise School Readiness Prediction System (ESRPS). The study focuses on creating a practical and effective machine learning-based system for predicting children's school readiness. The relevant R&D model is Borg and Gall's, which involves steps such as needs analysis to identify factors influencing school readiness, product development by integrating machine learning models like Decision Tree, Naive Bayes, Random Forest, and SVM, testing and validation through performance metrics like accuracy, precision, and recall, and implementation through a user-friendly web-based interface and database for practical use. This approach ensures the system is scientifically validated and practically applicable.

The research data is sourced from the administrative records of each child that were submitted to elementary schools in various cities in 2023. The suitable technique for selecting the sample in this study is stratified random sampling. This method ensures that the sample is representative of the population by dividing it into specific subgroups (e.g., cities: Surakarta, Samarinda, Surabaya) and randomly selecting participants from each subgroup. This approach is appropriate because it accounts for diversity in the dataset, such as regional differences, while maintaining balance across groups to ensure the results are generalizable and unbiased. The total sample consists of 300 students, with 100 students each from Surakarta, Samarinda, and Surabaya. The administrative data includes details such as the child's age, gender, father's education level, and mother's education level. This information serves as the foundation for the analysis in this study. The description of the variables is shown in Table 1.

Table1. Variable Description Table

Variable	Description	Value
Child's Age	The age of the child in years	5 years = 0 6 years = 1 7 years = 2

Gender	The gender of the child	Female = 1
		Male = 2

Father's Education Level	The highest level of education attained by the father	No Schooling = 0 Elementary/Middle School = 1 High School-Bachelor's Degree = 2
Mother's Education Level	The highest level of education attained by the mother	No Schooling = 0, Elementary/Middle School = 1 High School-Bachelor's Degree = 2

Modeling

The machine learning model development process begins with data preprocessing, which includes tasks such as handling missing values, encoding categorical variables, and normalizing or scaling features to prepare the data (Zhou & Song, 2020). The next step is parameter selection, where key parameters such as the criterion for splits, maximum depth, minimum samples split, minimum samples leaf, and the number of features considered for the best split are chosen. Once parameters are selected, the model is trained using various combinations of these parameters. After training, the model's performance is evaluated using metrics like accuracy, precision, recall, F1-score, and ROC-AUC score. To optimize performance, hyperparameter tuning is conducted using methods like Grid Search or

Ra
pe

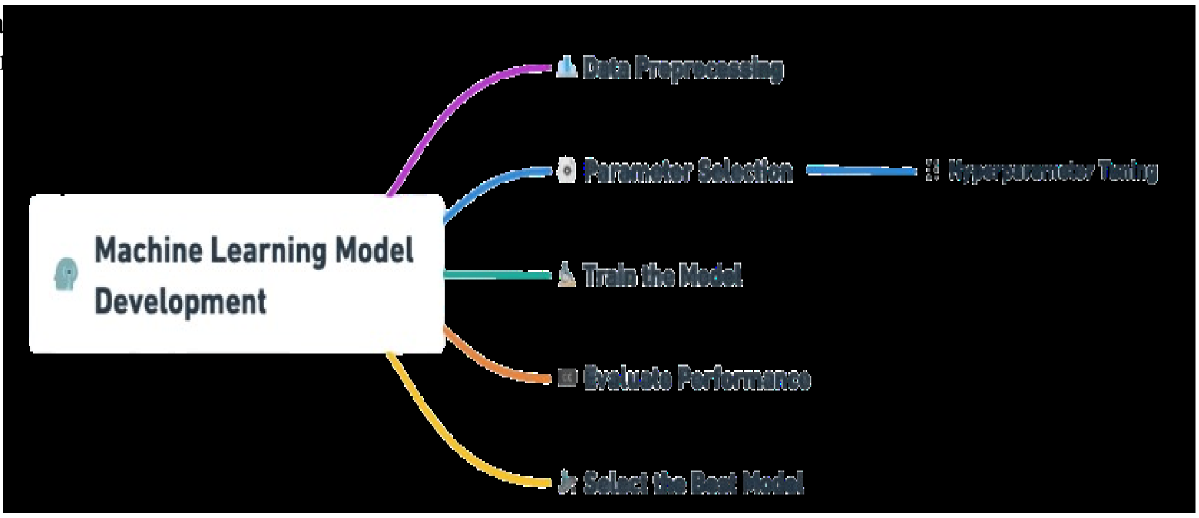


Figure 1. Machine Learning Model Development

Decision Trees

To train data using the Decision Trees algorithm and evaluate various parameter scenarios to identify the best-performing architecture (Ma & Zhou, 2018), several steps need to be followed. First, in data preprocessing, the dataset must be loaded and cleaned, handling any missing values, encoding categorical variables like gender and parental education, and normalizing or scaling the features if necessary. Next, during parameter selection, key factors impacting Decision Tree performance should be defined, such as the criterion used for splits (either Gini impurity or entropy for information gain), the maximum depth of the tree, the minimum number of samples required to split a node, the minimum number of samples required at a leaf node, and the number of features considered for the best split.

Once these parameters are set, the model is trained by using different combinations of these parameters to create multiple Decision Trees. The dataset is split into training and testing sets, and techniques like cross-validation are employed to evaluate the model's performance consistently. During the evaluation phase, various metrics are used to assess the model’s effectiveness, including accuracy (the percentage of correct predictions), precision, recall, F1-score (useful for imbalanced datasets), and ROC-AUC score (for binary classification problems).

In the hyperparameter tuning step, methods like Grid Search or Randomized Search are used to automate the process of finding the best parameter combinations. Cross-validation (e.g., 5-fold cross-validation) is recommended to prevent overfitting and ensure that the model generalizes well to unseen data. Finally, once all models have been trained and evaluated, the best-performing Decision Tree architecture is selected based on its performance on the validation data, allowing the identification of the optimal parameter configuration for the task at hand (Hamoud et al., 2018).

Table 2. Decision Trees Parameters

Parameter	Description	Possible Values
Criterion	The function to measure the quality of a split.	'Gini', 'entropy'
Max Depth	The maximum depth of the tree. Limits the number of splits in the tree to prevent overfitting.	None, 10, 20, 30, or any integer value
Min Samples Split	The minimum number of samples required to split an internal node.	2, 5, 10, or other integer values
Min Samples Leaf	The minimum number of samples required to be at a leaf node.	1, 2, 4, or other integer values

Max Features	The number of features to consider when looking for the best split.	None, 'sqrt', 'log2', or any fraction
Splitter	The strategy used to split at each node.	'best', 'random'
Max Leaf Nodes	The maximum number of leaf nodes in the tree. If None, an unlimited number of leaf nodes are allowed.	None, or any integer value
Min Weight Fraction Leaf	The minimum weighted fraction of the input samples required to be at a leaf node.	0.0, 0.1, or other float values

Support Vector Machine (SVM)

To implement training data using the Support Vector Machine (SVM) algorithm and evaluate different parameter scenarios, several steps are necessary. First, the dataset needs to be preprocessed by handling missing values, encoding categorical variables, and scaling or normalizing features, as SVMs are sensitive to feature scaling. Next, key parameters should be selected, including the type of kernel (linear, polynomial, radial basis function, or sigmoid), the regularization parameter (C), gamma (which controls the influence of each data point), and the degree for polynomial kernels. After selecting parameters, the model is trained using different combinations of these parameters, with the dataset split into training and testing sets. Cross-validation is used to evaluate performance consistently. The model's performance is then evaluated based on metrics like accuracy, precision, recall, F1-score, and ROC-AUC for binary classification. To optimize the model, hyperparameter tuning is conducted using Grid Search or Randomized Search, along with cross-validation, to avoid overfitting. Finally, the best-performing SVM model is selected based on its performance on validation data. The expected output of this process includes identifying the optimal parameter combination that delivers the best accuracy or performance metric, along with performance metrics that show how well the model performs on unseen data.

Table 3. Support Vector Machine (SVM) Parameters

Parameter	Description	Possible Values
Kernel	Specifies the kernel type to be used in the algorithm.	'linear', 'poly', 'rbf', 'sigmoid'
C (Regularization)	Controls the trade-off between classifying all training examples correctly and having a smooth decision boundary.	0.1, 1, 10, 100
Gamma	Defines how far the influence of a single training example reaches. Higher values lead to more focus on nearby points.	scale, auto, 0.1, 0.01, 0.001
Degree	Degree of the polynomial kernel function (only applicable for poly kernel).	2, 3, 4, 5

Naive Bayes

To implement training data using the Naive Bayes algorithm and evaluate different parameter scenarios, several steps are necessary. First, the dataset needs to be preprocessed by handling missing values and encoding categorical variables, as Naive Bayes requires numerical data. Feature scaling is generally not required for Naive Bayes since it calculates probabilities independently for each feature. Next, the appropriate type of Naive Bayes classifier should be chosen, such as Gaussian, Multinomial, or Bernoulli, depending on the nature of the data. For continuous features, the Gaussian Naive Bayes is suitable, while Multinomial is often used for discrete or count data, and Bernoulli is for binary data. After selecting the appropriate classifier, the model is trained using the preprocessed dataset, which is split into training and testing sets. Cross-validation is applied to ensure consistent performance evaluation across different subsets of the data. The model's performance is evaluated using metrics such as accuracy, precision, recall, F1-score, and ROC-AUC for binary classification problems. To optimize the model, parameter tuning is conducted. For Naive Bayes, this might involve adjusting parameters like `alpha` in the Multinomial or Bernoulli Naive Bayes classifiers, which is a smoothing parameter that helps handle zero probabilities. Techniques like Grid Search or Randomized Search, combined with cross-validation, can be used to identify the best parameter values and prevent overfitting. Finally, the best-performing Naive Bayes model is selected based on its performance on validation data. The expected outcome of this process is the identification of the optimal parameter configuration that yields the highest accuracy or other relevant performance metrics, along with a comprehensive set of metrics demonstrating how effectively the model performs on unseen data.

Table 4. Naive Bayes Parameters

Parameter	Description	Possible Values
alpha	Smoothing parameter used in Multinomial and Bernoulli Naive Bayes. Helps to handle zero frequencies in data.	Any non-negative float (e.g., 1.0, 0.5)
fit_prior	Indicates whether to learn class prior probabilities from the data.	True, False
class_prior	Allows specifying the prior probabilities of the classes. If not set, the model will estimate them from the data.	None, or an array of floats summing to 1
binarize	Threshold for converting numeric features into binary values, used in Bernoulli Naive Bayes.	Any float value (e.g., 0.0) or None
var_smoothing	Used in Gaussian Naive Bayes to add a small amount of variance to the calculation for numerical stability.	Any positive float (e.g., 1e-9)

Random Forest

To implement training data using the Random Forest algorithm and evaluate different parameter scenarios, several steps are necessary. First, the dataset needs to be preprocessed by handling missing values and encoding categorical variables. Although scaling or normalizing features is generally not required for Random Forest, it's important to ensure that all features are numeric. Next, key parameters should be selected, including the number of trees (`n_estimators`), maximum tree depth (`max_depth`), minimum samples to split a node (`min_samples_split`), minimum samples required at a leaf node (`min_samples_leaf`), and the number of features to consider for the best split (`max_features`). After selecting these parameters, the model is trained using different combinations of them, with the dataset split into training and testing sets. Cross-validation is used to ensure consistent performance evaluation across different subsets of the data. The model's performance is then evaluated based on metrics such as accuracy, precision, recall, F1-score, and ROC-AUC for binary classification tasks. To optimize the model, hyperparameter tuning is conducted using techniques such as Grid Search or Randomized Search combined with cross-validation. This helps identify the best parameter combinations while avoiding overfitting. Finally, the best-performing Random Forest model is selected based on its performance on validation data. The expected outcome of this process includes finding the optimal parameter combination that yields the highest accuracy or other relevant performance metrics. The results also include performance metrics that demonstrate how effectively the model performs on unseen data, ensuring that it generalizes well to new instances.

Table 5. Random Forest Model Parameters

Parameter	Description	Possible Values
n_estimators	The number of trees in the forest.	Any positive integer (e.g., 100, 200)
max_depth	The maximum depth of the tree.	None, or any positive integer (e.g., 10, 20)
min_samples_split	The minimum number of samples required to split an internal node.	Any positive integer (e.g., 2, 10)
min_samples_leaf	The minimum number of samples required to be at a leaf node.	Any positive integer (e.g., 1, 5)
max_features	The number of features to consider when looking for the best split.	‘auto’, ‘sqrt’, ‘log2’, or a float/int value
bootstrap	Whether bootstrap samples are used when building trees.	True, False
criterion	The function to measure the quality of a split.	‘gini’, ‘entropy’ (for classification); ‘mse’, ‘mae’ (for regression)
random_state	Controls the randomness of the bootstrapping of the samples used when building trees.	None, or any integer
max_leaf_nodes	The maximum number of leaf nodes in each tree.	None, or any positive integer (e.g., 10, 50)
min_impurity_decrease	A node will be split if this split induces a decrease of the impurity greater than or equal to this value.	Any non-negative float (e.g., 0.0, 0.1)
class_weight	Weights associated with classes for classification problems.	None, ‘balanced’, or a dictionary with class weights

Deployment

The Deployment process covers the entire cycle from data processing, model training, user interface creation, to storage and display of prediction results in interactive and real-time. Models stored in a specific format, then applied in the development of prediction web systems

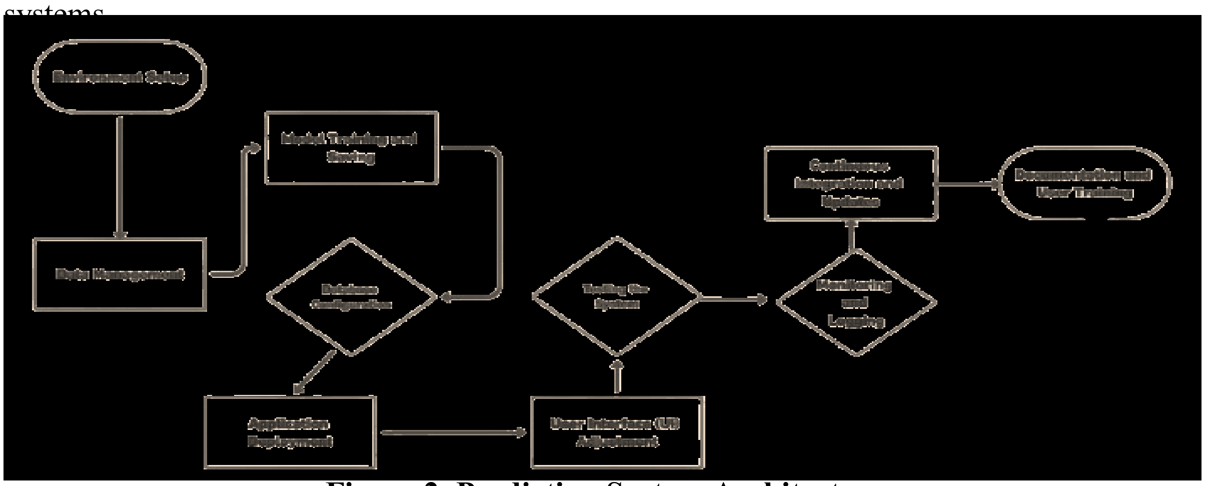

Environment Setup
Data Management
Model Training and Saving
Database Configuration
Application Deployment
User Interface (UI) Adjustment
Testing the System
Monitoring and Logging
Continuous Integration and Updates
Documentation and User Training

Figure 2. Prediction System Architecture

Results and Discussion

Model Performance Evaluation

The system was evaluated using four different machine learning algorithms: Decision Tree, Random Forest, Naive Bayes, and Support Vector Machine (SVM). The dataset was split into training and testing sets, with 80% used for training and 20% for testing. The accuracy scores for each model on the test set were as follows:

Table 6. Model Accuracies

Model	Accuracy
Decision Tree	55%
Random Forest	45%
Naive Bayes	55%
SVM	50%

These results show the performance of each algorithm when applied to the dataset. The Decision Tree and Naive Bayes models achieved the highest accuracy (55%), while SVM performed slightly lower (50%), and the Random Forest model had the lowest accuracy at 45%.

Database Implementation and Prediction Recording

The system stores prediction data in a SQLite database, which includes fields such as name, gender, age, father's education, mother's education, and the prediction result. This functionality allows for saving and retrieving historical predictions for future reference, providing users with a reliable log of all predictions made. By organizing the data in a structured manner, the system ensures easy access and filtering of records, enabling users to track the development of individual students or identify patterns over time. Additionally, the SQLite implementation is efficient for local applications, offering a lightweight solution that requires minimal configuration while still providing robust data storage capabilities. In the future, this setup can be scaled or migrated to a more powerful database system if needed, making it adaptable for larger datasets and multi-user environments.

User Interface Design

The user interface is implemented as a web-based application that allows users to input student details and select a prediction model. The interface displays fields for Name, Gender, Age, Father's Education, and Mother's Education, with dropdown menus for input to standardize the data and reduce errors. Users can select a prediction model from four options: Decision Tree, Random Forest, Naive Bayes, and SVM, ensuring flexibility and allowing them to choose the most suitable algorithm based on their needs. Once the prediction is made, the system displays the result clearly within the interface, enhancing user engagement and immediacy. Furthermore, users can access a dedicated section to view past predictions stored in the database, providing a convenient way to track and review historical data. The design is user-friendly, intuitive, and adaptable, allowing for easy updates and expansions to accommodate more features or data fields in the future.

Model Selection

The system supports four different machine learning models, allowing the user to select from Decision Tree, Random Forest, Naive Bayes, and SVM for school readiness prediction. Each model was trained using the preprocessed dataset, ensuring that they can accurately classify and predict outcomes based on the features provided by the user. By offering multiple models, the system provides flexibility, enabling users to choose the most appropriate algorithm depending on the characteristics of their data or specific requirements, such as simplicity, interpretability, or accuracy (Gardner & Brooks, 2018). The Decision Tree model is ideal for users who prefer a clear and interpretable decision-making process (Hamoud et al., 2018), while the Random Forest offers enhanced accuracy through an ensemble approach (Martinez Pedroso, 2016). Naive Bayes is efficient for cases with categorical data (Gupte et al., 2014), and SVM is suitable for binary classification tasks where a hyperplane separation can be leveraged (Shao et al., 2014). This variety of models enhances the system's adaptability, making it suitable for different datasets and scenarios in educational prediction contexts.

Error Handling

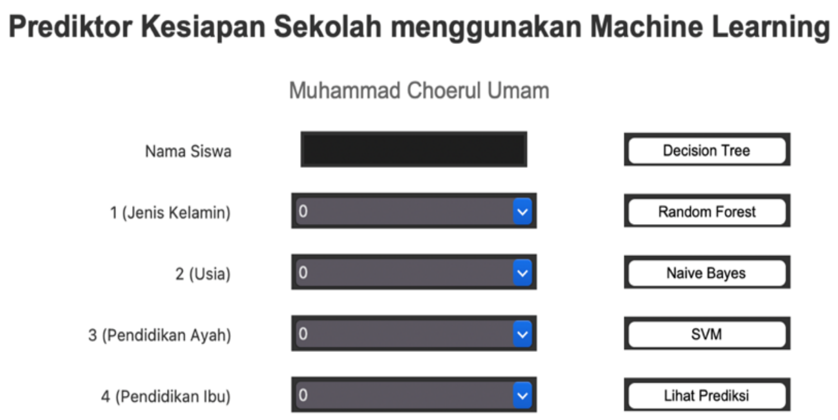
The system includes basic error handling. If incorrect input is provided, the system displays an error message in the UI, ensuring users are informed of any input-related issues and guiding them to correct their entries. This approach helps maintain the stability and usability of the application, preventing crashes or unexpected behavior that could disrupt the user experience. The error messages are designed to be clear and user-friendly, offering specific guidance, such as indicating which field requires attention or suggesting the correct format for input values. In future updates, the system could be enhanced with real-time validation that checks inputs as users type, further reducing errors and providing immediate feedback to improve data accuracy and streamline the prediction process.

Discussion

First, Model Performance Evaluation, Based on the accuracy results, the Decision Tree and Naive Bayes models performed best, both achieving an accuracy of 55%. This suggests that these models may be more suitable for the dataset used in this study, possibly due to their simplicity and ability to handle categorical data effectively(Wibowo & Oesman, 2020). The SVM model achieved an accuracy of 50%, indicating that it might require parameter tuning or feature scaling to enhance its performance further. The Random Forest model, despite being an ensemble method known for improving accuracy through multiple decision trees, had the lowest accuracy at 45%. This may suggest that the dataset size or feature complexity did not benefit as much from the ensemble approach as expected.

Second, Database Implementation using SQLite for prediction storage ensures that results are saved locally, providing users with a simple and effective solution for managing historical data. For larger applications or multi-user environments, a more scalable database solution such as PostgreSQL or MySQL might be more appropriate to handle increased data volume and concurrent access(Martinez Pedroso, 2016).

Third, User Interface Design The web-based interface provides a user-friendly and intuitive experience, allowing users to interact with the system directly from a browser. The design offers dropdown menus for user input, ensuring that data entries are standardized and reducing input errors. This interface improves the user experience compared to the previous desktop-based application developed with Tkinter. However, expanding this interface further could involve integrating more detailed input validation mechanisms and optimizing the layout for various screen sizes to enhance usability on different devices, including mobile phones and tablets.



Prediktor Kesiapan Sekolah menggunakan Machine Learning

Muhammad Choerul Umam

Nama Siswa	
1 (Jenis Kelamin)	0
2 (Usia)	0
3 (Pendidikan Ayah)	0
4 (Pendidikan Ibu)	0

Figure 3, User Interface Design The web-based interface

Fourth, Model Selection The system's flexibility in allowing users to choose between four different models provides versatility. Users can select the model that best fits their dataset characteristics and prediction needs. To enhance this feature, incorporating a hyperparameter tuning process (e.g., GridSearchCV or RandomizedSearchCV) could further optimize each model's performance. Additionally, integrating an intelligent model recommendation system based on the user's data could make the system more efficient and user-friendly.

Fifth, Error Handling and System Stability The error-handling mechanism ensures that the system remains stable and user-friendly, as incorrect inputs do not crash the application. This feature is critical for maintaining a smooth user experience. With the web-based interface, error messages can be displayed directly on the screen, guiding users to correct their inputs. In future developments, incorporating real-time input validation and providing specific error messages could further enhance system stability.

Sixth, Scalability and Future Enhancements The transition to a web-based system, as depicted in the provided image, improves accessibility and scalability. Users can access the system remotely through a web browser, making it more versatile. To further enhance scalability, the system could be deployed on cloud platforms such as AWS or Heroku, enabling it to handle larger datasets and multiple concurrent users, providing real-time predictions and a responsive user experience. Additionally, implementing a responsive design that adapts to various screen sizes would allow the system to be used effectively on mobile devices, expanding its accessibility further.

The results of this study have both conceptual and practical implications. Conceptually, the research demonstrates the feasibility of using machine learning algorithms, such as Decision Tree, Naive Bayes, Random Forest, and SVM, to assess school readiness, contributing to the growing field of educational data mining. It highlights the importance of data preprocessing and algorithm selection in achieving optimal model performance, emphasizing that simpler models may be more effective for specific datasets. Practically, the study offers a valuable tool for educators and parents to assess children's readiness for elementary school, enabling tailored interventions to support their transition. The development of a web-based application ensures ease of access and real-time usability, making the system adaptable to diverse educational settings. This system also provides a scalable solution for storing and analyzing readiness data, aiding in long-term tracking and decision-making. These outcomes enhance the ability to support children's educational development effectively.

Conclusion

The Enterprise School Readiness Prediction System (ESRPS) demonstrates the potential of utilizing machine learning algorithms to assess children's readiness for elementary education. By employing models such as Decision Tree, Random Forest, Naive Bayes, and SVM, the system provides a practical approach to evaluating readiness based on demographic and developmental factors. Despite the modest accuracy rates of 55% achieved by the Decision Tree and Naive Bayes models, the study highlights the importance of data preprocessing, model training, and hyperparameter tuning in optimizing performance. Furthermore, the deployment of ESRPS as a web-based application ensures user accessibility and real-time prediction capabilities, offering a scalable solution for educators and parents. Future work should focus on increasing dataset size, refining feature extraction, and integrating additional models to improve the system's accuracy and adaptability in diverse educational settings.

Recommendation

Policymakers should consider integrating data-driven tools like the ESRPS into educational planning and evaluation frameworks to enhance decision-making processes. They should support the development and deployment of machine learning systems for assessing school readiness by allocating resources for large-scale data collection and system training while ensuring ethical standards for handling children's data. For teachers, the ESRPS can be a valuable tool for identifying students' strengths and areas needing support before entering elementary school, enabling personalized teaching strategies and interventions. Professional development programs should be conducted to help teachers understand the system's features and interpret its predictions effectively. For further researchers, expanding the dataset size and diversity is crucial to improving the accuracy and generalizability of the models. Future studies should explore hybrid machine learning approaches, refine feature extraction techniques, and conduct longitudinal research to evaluate the system's long-term effectiveness and its impact on students' educational outcomes. Additionally, research on its applicability across different cultural and educational contexts would ensure broader relevance.

References

Al Mayahi, K., & Al-Bahri, D. M. (2020). Machine Learning Based Predicting Student Academic Success. *International Congress on Ultra Modern Telecommunications and Control Systems and Workshops*, *2020-October*. https://doi.org/10.1109/ICUMT51630.2020.9222435

Cattell, L., & Bruch, J. (2021). Identifying Students at Risk Using Prior Performance versus a Machine Learning Algorithm. REL 2021-126. In *Regional Educational Laboratory Mid-Atlantic*.

Dockett, S., & Perry, B. (2002). Who's Ready for What? Young Children Starting School. *Contemporary Issues in Early Childhood*, *3*(1). https://doi.org/10.2304/ciec.2002.3.1.9

Gardner, J. P., & Brooks, C. (2018). Evaluating Predictive Models of Student Success: Closing the Methodological Gap. *Journal of Learning Analytics*, *5*(2). https://doi.org/10.18608/jla.2018.52.7

Gill, C. K., Vig, D., & Chawla, A. (2020). The Developmental Readiness of Government School Teachers. *Journal of Education, Society and Behavioural Science*. https://doi.org/10.9734/jesbs/2020/v33i330208

Gupte, A., Joshi, S., Gadgul, P., & Kadam, A. (2014). Comparative Study of Classification Algorithms used in Sentiment Analysis. *(IJCSIT) International Journal of Computer Science and Information Technologies*, *5*(5).

Halmatov, M. (2018). Assessment of Psychological Readiness Situation of Students Starting to Primary School. *International Education Studies*, *11*(5). https://doi.org/10.5539/ies.v11n5p85

Hamoud, A. K., Hashim, A. S., & Awadh, W. A. (2018). Predicting Student Performance in Higher Education Institutions Using Decision Tree Analysis. *International Journal of Interactive Multimedia and Artificial Intelligence*, *5*(2). https://doi.org/10.9781/ijimai.2018.02.004

Jannah, M. (2023). Perceptions of Preschool Teachers on Children's School Readiness in Purwakarta Regency. *International Social Sciences and Humanities*, *2*(2). https://doi.org/10.32528/issh.v2i2.264

Kokkalia, G., Drigas, A., Economou, A., & Roussos, P. (2019). School readiness from kindergarten to primary school. *International Journal of Emerging Technologies in Learning*, *14*(11). https://doi.org/10.3991/IJET.V14I11.10090

Lakkaraju, H., Aguiar, E., Shan, C., Miller, D., Bhanpuri, N., Ghani, R., & Addison, K. L. (2015). A machine learning framework to identify students at risk of adverse academic outcomes. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *2015-August*. https://doi.org/10.1145/2783258.2788620

Ma, X., & Zhou, Z. (2018). Student pass rates prediction using optimized support vector machine and decision tree. *2018 IEEE 8th Annual Computing and Communication Workshop and Conference, CCWC 2018*, *2018-January*. https://doi.org/10.1109/CCWC.2018.8301756

Martinez Pedroso, P. B. (2016). High Availability and Load Balancing for Postgresql Databases : Designing and Implementing. *International Journal of Database Management Systems*, *8*(6). https://doi.org/10.5121/ijdms.2016.8603

Nuranisah, Efendi, S., & Sihombing, P. (2020). Analysis of algorithm support vector machine learning and k-nearest neighbor in data accuracy. *IOP Conference Series: Materials Science and Engineering*, *725*(1). https://doi.org/10.1088/1757-899X/725/1/012118

NURLINA, I. I. , & S. (2019). Pre Test Prediction System for Preparing Readiness for Basic Education. . *Seminar Nasional Komunikasi Dan Informatika*, 119–124.

Pekdogan, S., & Akgul, E. (2016). Preschool Children's School Readiness. *International Education Studies*, *10*(1). https://doi.org/10.5539/ies.v10n1p144

Pregowska, A., & Osial, M. (2021). What Is An Artificial Neural Network And Why Do We Need It? *Frontiers for Young Minds*, *9*. https://doi.org/10.3389/frym.2021.560631

Shao, Y. H., Chen, W. J., & Deng, N. Y. (2014). Nonparallel hyperplane support vector machine for binary classification problems. *Information Sciences*, *263*. https://doi.org/10.1016/j.ins.2013.11.003

Shaw, D. S., Mendelsohn, A. L., & Morris, P. A. (2021). Reducing Poverty-Related Disparities in Child Development and School Readiness: The Smart Beginnings Tiered Prevention Strategy that Combines Pediatric Primary Care with Home Visiting. *Clinical Child and Family Psychology Review*, *24*, 669–683. https://api.semanticscholar.org/CorpusID:237468561

Wibowo, A. H., & Oesman, T. I. (2020). The comparative analysis on the accuracy of k-NN, Naive Bayes, and Decision Tree Algorithms in predicting crimes and criminal actions in Sleman Regency. *Journal of Physics: Conference Series*, *1450*(1). https://doi.org/10.1088/1742-6596/1450/1/012076

Zhou, Y., & Song, Z. (2020). Effectiveness analysis of machine learning in education big data. *Journal of Physics: Conference Series*, *1651*(1). https://doi.org/10.1088/1742-6596/1651/1/012105