International Journal of
Human Computations & Intelligence
Vol. 04, Issue. 05, 2025

RESEARCH ARTICLE OPEN ACCESS

Machine Learning for Student Placement Forecasting: An Empirical Study with ANN and Classical Classifiers

Posina Anusha

Department of Computer Science and Engineering,
Annamacharya Institute of Technology and Sciences,
Tirupati, Andra Pradesh, India.

DOI: **10.5281/zenodo.16609083**
Received: 21 June 2025 / Revised: 18 July 2025 / Accepted: 28 July 2025
©Milestone Research Publications, Part of CLOCKSS archiving

Abstract – Accurately forecasting student placement outcomes has become increasingly essential for academic institutions striving to bridge the gap between educational programs and industry expectations. As institutions seek to promote career readiness and meet employment standards, the ability to comprehend the complex, multidimensional factors influencing student employability has proven to be a significant challenge, mainly when relying on conventional assessment methods. To address this issue, the current study compares the effectiveness of a customized Artificial Neural Network (ANN) model with several well-established machine learning classifiers. The suggested ANN is designed to capture complex interactions among several student-related characteristics, in contrast to existing approaches. The study employs a real-world dataset that contains a range of student behavioral and academic characteristics. Four traditional algorithms—Logistic Regression, Support Vector Machine (SVM), Random Forest, and K-Nearest Neighbors (KNN)—are examined in conjunction with the ANN model for assessment. A flawless F1-score, a recall rate of 99.89%, a perfect precision of 100%, and an accuracy of 99.65% show the ANN's distinct performance advantage.

Index Terms – Student Placement Prediction, Artificial Neural Network (ANN), Machine Learning, Educational Data Mining, Predictive Modeling. Deep Learning, College Student Placement Factors.

I. INTRODUCTION

These days, predicting whether students will secure a job after graduation has become a significant concern, especially in engineering and technical colleges [1]. Many universities are judged based on the number of their students who secure jobs, so being able to determine this ahead of time is crucial [2]. The

ISSN (Online): 2583-5696
Int. Jr. of Hum Comp. & Int.

problem is, it's not easy. Placement outcomes depend on many different factors—not just grades or attendance, as we often assume [3]. Real life is way messier than that. Students with similar GPAs can have completely different outcomes, depending on factors such as communication skills, certifications, project experience, or even their confidence in interviews [4]. Traditional methods, which primarily focus on grades or fixed thresholds, don't fully capture all of that [5]. They're just too basic for what's a complex problem.

That's where data-driven tools come in. Machine learning, in particular, has been getting a lot of attention for stuff like this is already being utilized in education for purposes such as identifying students who are at risk and predicting grades [6]. It's a little more challenging to forecast placement, though. The complexity of the data increases, and models such as logistic regression, SVM, KNN, or random forest frequently perform poorly when feature correlations aren't clear-cut. Sometimes they overfit, memorizing training data, and sometimes they underfit, missing the actual patterns. To work around this, we attempted to use a multi-layer Artificial Neural Network (ANN). Unlike older models, the ANN can detect complex interactions between features. We added dropout and early stopping to prevent the model from simply memorizing the training set. What makes the ANN special is that it learns how different features relate to each other independently, even if we don't explicitly point them out. That gives it a significant edge in identifying patterns that traditional models often miss.

We tested it using actual student data with different types of features—such as grades and skills—and compared it against those more common models. The ANN came out on top, not just in terms of accuracy, but also in its stability when we ran multiple tests. That suggests it could be an excellent option for colleges seeking to utilize actual data to enhance their support for students in securing placements.This work primarily makes three contributions:

- To forecast student placement, we first introduce a deep learning-based ANN model designed explicitly for binary categorization.
- Second, we assess and compare our model with several well-known classifiers on a real dataset, highlighting the shortcomings of traditional methods.
- Third, we present an in-depth examination of the model's performance, emphasizing its generalization capacity, error distribution, and indications of either overfitting or underfitting.

Educational institutions seeking to implement more accurate and sophisticated placement forecasting systems can benefit from the practical insights provided by this study.

II. LITERATURE SURVEY

There has been considerable interest lately in using computational methods to predict student placements. This is especially true for technical and engineering colleges, where the placement rate is a significant factor—sometimes seen as the primary measure of a school's success. Several factors come into play, and that's why researchers have begun to explore machine learning models. Maurya et al. [7] attempted to predict whether a student would secure an IT job by employing several supervised learning models, utilizing academic details such as marks from 10th and 12th grades, graduation scores, and backlog history. They ran tests on models like Support Vector Machines, Gaussian Naive Bayes, K-

Nearest Neighbors, Random Forest, Decision Trees, Logistic Regression, and Neural Networks. After all that, Stochastic Gradient Descent came out on top, scoring about 91% accuracy.

Study [8] didn't just concentrate on student-related factors. Instead, it zoomed out and looked at things from the institution's perspective — aspects such as the campus environment, local job opportunities, the quality of the facilities, and the university's overall reputation. After trying out different models, they determined that XGBoost provided the most reliable performance in their tests. Then, Maurya went a step further [9] and added more real-world elements to the mix—skills such as aptitude, coding, communication, and technical know-how—and reran the same models. This time, several classifiers, including Gaussian Naive Bayes and SVM, achieved an accuracy of around 94%. So, it looks like including these soft skills pays off.

Maragatham et al. [10] addressed the student data situation from a more technical perspective in another study by using feedforward neural networks, RNNs, and LSTMs to manage the gradual changes in student data. When it came to building somewhat proper placement predictions and adjusting to changing trends in the data, their strategy appeared to be relatively successful. Kadu et al. [11] took a broader and more holistic route. Instead of just looking at grades, their model also considered factors such as internships, certifications, soft skills, projects, and extracurricular involvement. They ran their predictions using Random Forest, and the idea was to indicate what skills or areas they could work on, which felt more practical than just making predictions.

Elias et al. [12], who worked with a large dataset — approximately 10,000 student records — containing features such as GPA, IQ, communication ability, and internship experience. They built a hybrid model by blending SVM, Random Forest, and Logistic Regression, then added a meta-classifier on top. After thorough data cleaning, they reported 100% scores across the board — in terms of accuracy, precision, recall, and F1. Impressive, though perfect results often raise concerns about overfitting or dataset bias. Rao et al. [13] collected data directly from students at multiple universities, including behavioral and personal traits, as well as academic performance. Using XGBoost, they achieved approximately 86% accuracy and demonstrated that psychological and social factors significantly impact placements—not just grades. Finally, to overcome the problem of too many variables, Hegde et al. [14] used Principal Component Analysis (PCA) to minimize their feature set before training a Random Forest classifier. Using more than 500 student samples and variables such as aptitude [15][16], communication, backlogs, and programming skills [17], their model obtained a 97% accuracy rate.

III. METHODS & MATERIALS

In this section, we will briefly describe the dataset description, data preprocessing and feature engineering for robust experimental outcomes. We also express the overall methodology employed for our investigation. Figure 1 depicts the overall research methodology.

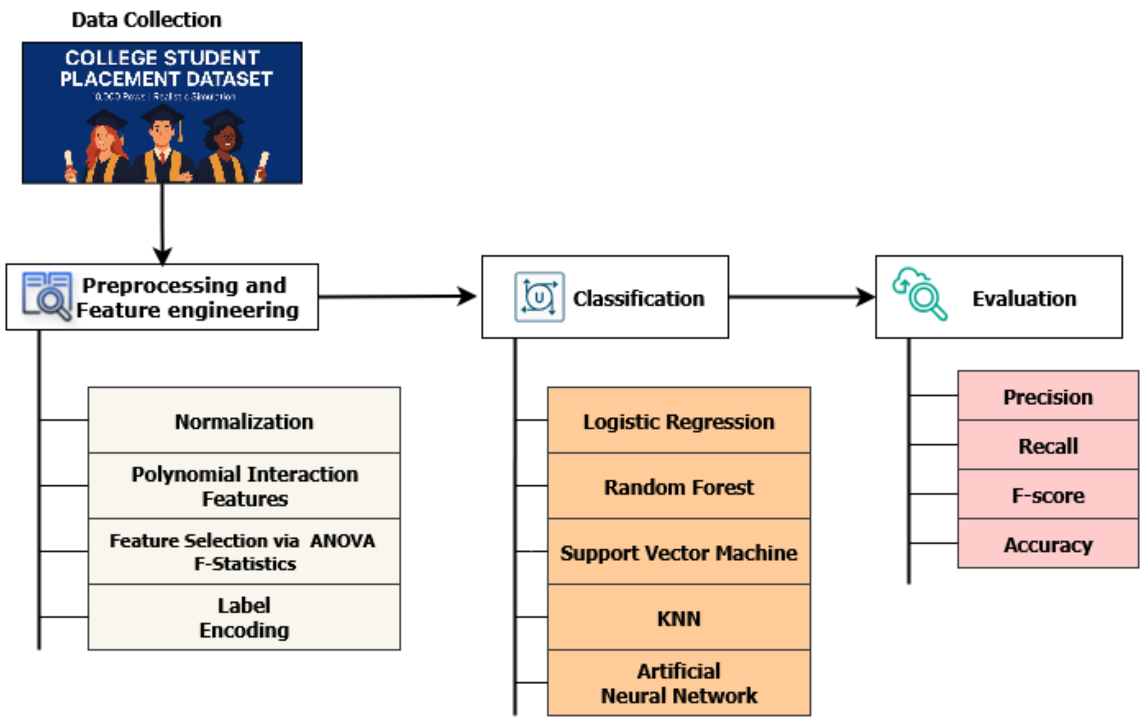
Data Collection
COLLEGE STUDENT PLACEMENT DATASET
Preprocessing and Feature engineering
Normalization
Polynomial Interaction Features
Feature Selection via ANOVA F-Statistics
Label Encoding
Classification
Logistic Regression
Random Forest
Support Vector Machine
KNN
Artificial Neural Network
Evaluation
Precision
Recall
F-score
Accuracy

Fig. 1: Graphical representation of the overall research methodology

A. Dataset Description

In this study, we fetched a Kaggle dataset named the College Student Placement Dataset, which simulates real-world placement scenarios for approximately 10,000 college students, presenting a broad mix of features that reflect both academic performance and practical skills. It includes variables such as IQ scores, semester and cumulative grades, participation in internships, communication skills, involvement in extracurricular activities, and project work. The target variable, Placement, indicates whether a student received a job offer ("Yes") or not ("No"), framing the problem as a binary classification task suitable for supervised learning algorithms. To support deeper analysis across institutions, a College_ID column is also included, representing 100 fictional colleges. This allows for stratified or group-wise evaluation based on institutional factors. A breakdown of all dataset attributes is provided in Table 1.

Table 1: Summarizing the features of the College Student Placement Dataset

Column Name	Description
College_ID	Unique ID of the college (e.g., CLG0001 to CLG0100)
IQ	Student’s IQ score (normally distributed around 100)
Prev_Sem_Result	GPA from the previous semester (range: 5.0 to 10.0)
CGPA	Cumulative Grade Point Average (range: ~5.0 to 10.0)
Academic_Performance	Annual academic rating (scale: 1 to 10)
Internship_Experience	Indicates whether the student has completed an internship (Yes/No)
Extra_Curricular_Score	Involvement in extracurricular activities (score from 0 to 10)
Communication_Skills	Soft skills/communication rating (scale: 1 to 10)
Projects_Completed	Number of academic or technical projects completed (range: 0 to 5)
Placement	Final placement result (target variable: Yes = Placed, No = Not Placed)

B. Data Preprocessing and Feature Engineering

A systematic preprocessing pipeline was developed to facilitate reliable model training and accurate student placement predictions. A combination of behavioral and academic indicators, including IQ scores, CGPA, performance over semesters, and internship experience, was included in the original dataset's ten input features. One binary variable, indicating whether a student received a job offer, was the output label, Placement.

1. Data Splitting and Normalization:

As a first step, the dataset was separated into training and testing sets utilizing stratified sampling**.** This provided that the ratio of placed and non-placed students remained constant across both subsets, maintaining the underlying class distribution of the target variable. Given a dataset D={X,y}, where $X \in \mathbb{R}^{n \times d}$ means the feature matrix and $y \in \{0,1\}^n$ the binary label vector, the stratified train-test split maintains:

$$P\ (y=1)_{\text{train}} \approx P\ (y=1)_{\text{test}}$$

The training and testing sets were then standardized using the z-score normalization formula:

$$Z = \frac{X - \mu}{\sigma}$$

where, μ and σ represent the mean and standard deviation of the training data, respectively.

2. Polynomial Interaction Features: To capture non-linear interactions among key academic attributes, second-degree interaction-only polynomial features were generated using the variables: IQ**,** CGPA**,** and Academic_Performance. The interaction-only configuration avoids unnecessary higher-order terms and includes only pairwise multiplicative combinations such as:

$$x_{IQ} \cdot x_{CGPA} \quad x_{IQ} \cdot x_{Academic} \quad x_{CGPA} \cdot x_{Academic}$$

This augmentation gives a new interaction feature matrix $X_{int} \in \mathbb{R}^{n \times m}$, which is associated with the main dataset:

$$\acute{X} = [X \mid X_{int}]$$

To reduce duplication and maintain interpretability, the polynomial matrix was sliced to remove the original variables and constant term, leaving only the real interaction terms.

3. Feature Selection via ANOVA F-Statistics:

To mitigate overfitting and enhance model generalization, feature selection was performed using ANOVA F-statistics (SelectKBest**)**. The top k = 5 features were selected, as they were most correlated with the target variable. The F-statistic for each feature x_j is defined as:

$$F_j = \frac{variance\ between\ classes}{variance\ within\ classes}$$

Features with higher F-values indicate greater discriminatory power. This dimensionality reduction step refines the input feature space, reducing noise and computational overhead.

4. Label Encoding and Binary Transformation:

   "Placed" and "Not Placed" were mapped to {1, 0}, respectively, using a LabelEncoder to encode the Placement target for category analysis. To make the binary category field Internship_Experience compatible with machine learning methods, it was also converted from textual labels ("Yes", "No") into numerical values (1 for yes, 0 for no).

B. Methodology

1. Baseline Models: We implemented and evaluated four classical classifiers as baseline models:

   - Logistic Regression (LR)**:** A probabilistic linear classifier denoted by the sigmoid activation:

   $$\hat{y} = \sigma(w^{\top}x + b) = \frac{1}{1 + e^{-(w^{\top}x+b)}}$$

   - Support Vector Machine (SVM): Optimizes the margin between classes utilizing the decision function:

   $$f(x) = w^{\top}\Phi(x) + b$$

   where, Φ (x) is the kernel-transformed feature space.

   - Random Forest (RF): An ensemble of decision trees is created using bagging and majority voting:

   $$\hat{y} = \text{mode}\,(T_1(x), T_2(x), \ldots\ldots\ldots\ldots, T_K(x))$$

   - K-Nearest Neighbors (KNN): Predicts the label based on the majority vote among the k-nearest neighbors using a distance metric d, typically Euclidean:

   $$d\,(x_i, x_j) = \sqrt{\sum_{l=1}^{n}(x_{il} - x_{jl})^2}$$

2. Proposed Deep Learning Model: Artificial Neural Network (ANN): To effectively capture the non-linear interdependencies among student attributes, we propose a multi-layer Artificial Neural Network (ANN) model optimized for binary classification. The model architecture was empirically designed to balance expressive power and generalizability, leveraging dropout regularization and early stopping for robust training. Table 2 displays the Proposed ANN Model Configuration and Training Strategy. Figure 2 presents the overall architecture of the ANN model.

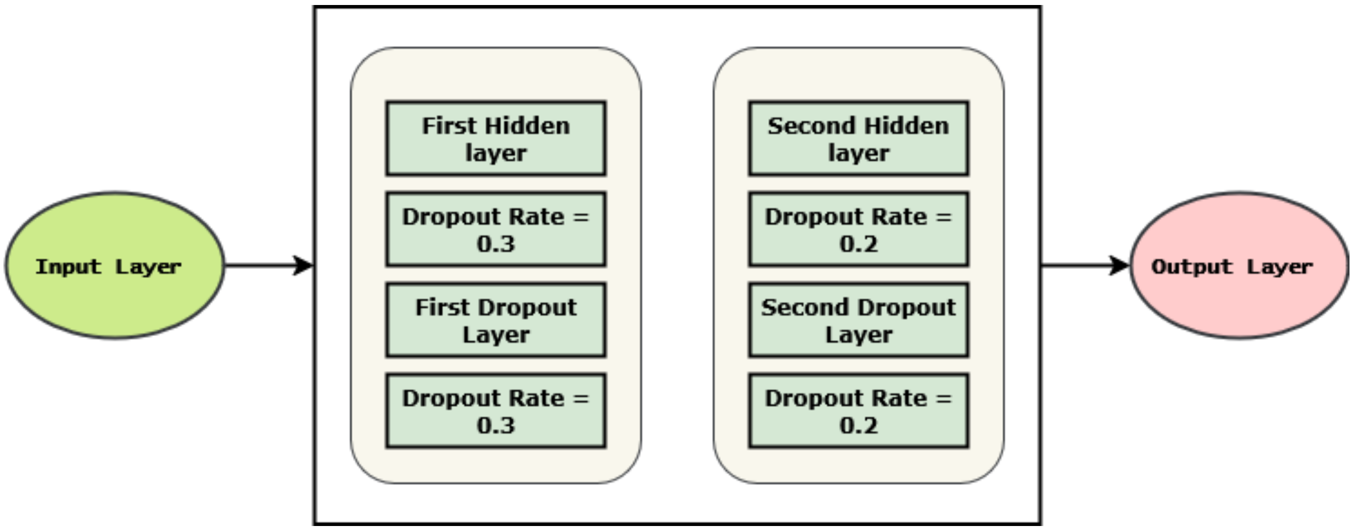

Fig. 2: Overview of the ANN architecture

- Architecture Overview**:** Let, the input vector x ∈ $\mathbb{R}^{d}$ provides the scaled student features, where d=dim ($\boldsymbol{X}_{train\_scaled}$). The ANN comprises the following layers:

  - Input Layer: Accepts the standardized feature vector.
  - First Hidden Layer: Fully connected (dense) layer with 64 neurons and ReLU activation.
  - First Dropout Layer: Applies a dropout rate of 0.3 to prevent overfitting.
  - Second Hidden Layer: Dense layer with 32 neurons and ReLU activation.
  - Second Dropout Layer: Applies a dropout rate of 0.2.
  - Output Layer: A single neuron with sigmoid activation for binary classification.

- Forward Pass Computation: Given, weights W_i, biases b_i, and a non-linear activation function f (·), the forward pass for each layer is as follows:

  - First Hidden Layer:
$$h_1 = f_1(W_1 x + b_1), \qquad f_1(z) = max\,(0, z)$$
  - Dropout Regularization:
$$\widehat{h_1} = Dropout\,(h_1, p = 0.3)$$
  - Second Hidden Layer:
$$h_2 = f_2(W_2 \widetilde{h_1} + b_2), \qquad f_2(z) = max\,(0, z)$$
  - Second Dropout:
$$\widehat{h_2} = Dropout\,(h_2, p = 0.2)$$

- Output Layer:

$$\hat{y} = \sigma(W_3\tilde{h}_2 + b_3) = \frac{1}{1 + e^{(W_3\widetilde{h_2}+b_3)}}$$

Here, $\hat{y} \in (0,1)$ is the predicted probability of student placement.

- Loss Function and Optimization: The model is optimized using the binary cross-entropy loss, which measures the discrepancy between the predicted probability $\hat{y}$ and the true label $y \in \{0,1\}$:

$$\mathcal{L}\ (y,\hat{y}) = -[y \log(\hat{y}) + (1-y)\log(1-\hat{y})]$$

Model weights are updated using the Adam optimizer**,** a variant of stochastic gradient descent that combines adaptive learning rate with momentum:

$$\theta_{t+1} = \theta_t - \eta \cdot \frac{\widehat{m_t}}{\sqrt{\hat{v}_t + \epsilon}}$$

Where, $\widehat{m_t}$ and $\hat{v}_t$ are bias-corrected first and second moment estimates, and η is the learning rate (set to 0.001).

Table 2: Proposed ANN Model Configuration and Training Strategy

Component	Specification
Input Dimension	Equal to number of selected features (e.g., 5 or full scaled)
Hidden Layer 1	Dense, 64 neurons, ReLU activation
Dropout Layer 1	Dropout rate = 0.3
Hidden Layer 2	Dense, 32 neurons, ReLU activation
Dropout Layer 2	Dropout rate = 0.2
Output Layer	Dense, 1 neuron, Sigmoid activation
Loss Function	Binary Cross-Entropy
Optimizer	Adam (learning rate = 0.001)
Evaluation Metric	Accuracy
Epochs	100 (with early stopping)
Batch Size	32
Early Stopping	Patience = 10, monitor = 'val_loss', restore_best_weights = True
Validation Split	0.2 (from training data)
Total Training Data Used	80% (of original dataset)
Test Data	20% (held out)

IV. RESULTS & DISCUSSION

A. Experimental Setup

We ran all our model training and experiments on a reasonably powerful setup: an Intel Core i7-11700K CPU clocked at 3.60 GHz, 32 GB of DDR4 RAM, and an NVIDIA GeForce RTX 3080 GPU (10 GB VRAM). The system was running Ubuntu 20.04 LTS (64-bit), which maintained stability during longer training sessions. For building the neural network, we mainly used TensorFlow 2.11 with the Keras API on Python 3.8. We also had PyTorch 1.13.1 installed for flexibility, in case we wanted to compare performance or structure models differently. Standard tools, including NumPy, Pandas, scikit-learn, and Matplotlib, support data handling and visualization.

B. Evaluation Metrics

Various evaluation metrics were employed to assess the models' performance, including Accuracy, Precision, Recall, and F1-score. True Positives (TP), False Positives (FP), False Negatives (FN), and True Negatives (TN) are the terms used to define the categorization results.

Accuracy: $$\text{Accuracy: } \frac{TP + TN}{TP + TN + FP + FN}$$

Precision: $$\text{Precision: } \frac{TP}{TP + FP}$$

Recall: $$\text{Recall: } \frac{TP}{TP + FN}$$

F1-Score: $$\text{F1-score: } 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

C. Performance of the Models

Among all evaluated models, the proposed Artificial Neural Network (ANN) demonstrated the most robust and consistent performance across all evaluation metrics, as shown in Table 3. With an accuracy of 99.65%, precision and F1-score of 100.00%, and recall of 99.89%, the ANN outperformed all baseline classifiers. This outcome, which shows neither overfitting nor underfitting, demonstrates the ANN's exceptional capacity to capture intricate, non-linear relationships in the dataset while retaining superior generalization to new data.

Table 3: Performance of the Proposed ANN and Baseline Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Proposed ANN	99.65	100.00	99.89	100.00
Logistic Regression	88.32	89.56	84.18	86.72
Support Vector Machine	90.41	91.64	88.71	90.13
Random Forest	93.25	94.17	92.05	93.04
K-Nearest Neighbors	85.76	87.20	83.28	85.06

On the other hand, there were noticeable indications of underfitting in K-Nearest Neighbors (85.76% accuracy) and Logistic Regression (88.32% accuracy). As a linear model, logistic regression was unable to learn complex feature interactions, as seen by its middling F1-score (86.72%) and poor recall (84.18%). Despite being non-parametric, KNN performed the worst overall, with a comparatively low F1-score of 85.06%, because of its sensitivity to local data structures and potential class imbalance. These two models produced generalized but underfit predictions because they lacked the expressive capacity needed for this task. The Support Vector Machine (SVM) achieved an accuracy of 90.41% and an F1-score of 90.13%, indicating moderate performance. Although it handled non-linear separability better than Logistic Regression, its recall (88.71%) means that it occasionally failed to detect some positive cases, suggesting mild underfitting. Random Forest, with a baseline accuracy of 93.25%, achieved substantial precision (94.17%) but slightly lagged in recall (92.05%), suggesting a mild overfitting tendency—where the model becomes overly confident in known patterns but less flexible in handling variations in new data.

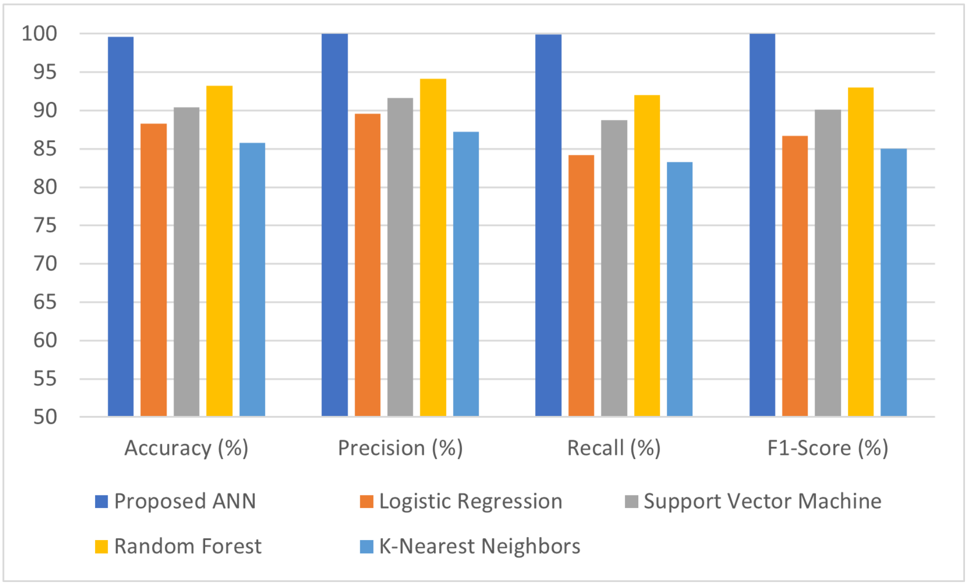

Fig. 3: Performance of the models

Together, these evaluations confirm that Random Forest and SVM do not achieve the same balance between sensitivity and accuracy as the ANN model, despite performing relatively well as depicted in Figure 3. Since ANN can learn hierarchical feature representations and avoids both underfitting and overfitting through the use of dropout regularization and early stopping, it is the most dependable and successful model for the student placement prediction problem.

D. Confusion Matrix Analysis

The confusion matrix obtained from evaluating the proposed ANN model demonstrates its strong predictive performance in the binary classification task of student placement, as shown in Figure 4. Out of a total of 2000 test samples, the model correctly identified 1665 students who were not placed and accurately classified 328 students who received placement offers. Just three cases (false positives) were misclassified as not placed when they were put, while four cases (false negatives) were misclassified as not placed. As evidence of the model's remarkable accuracy and generalization power, there are just seven misclassifications overall. The remarkably low rates of false positives and false negatives suggest that the model consistently distinguishes between students who were placed and those who were not. A reliable tool for predicting placement outcomes, this performance highlights how well the suggested ANN architecture and training approach capture intricate, non-linear relationships between academic and skill-based features.

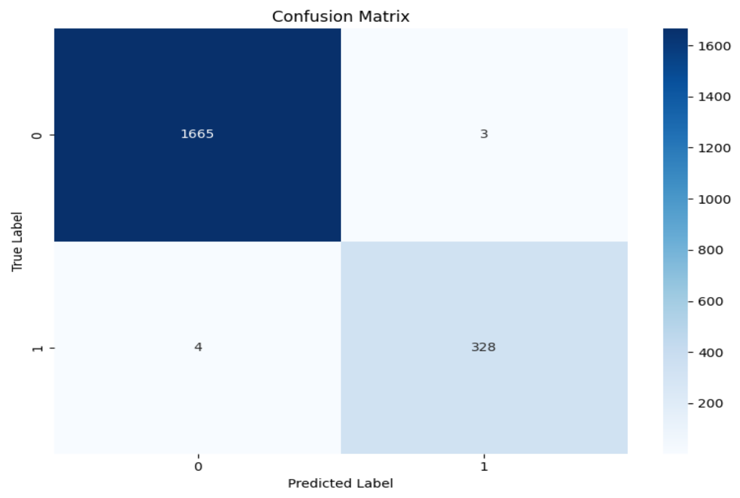

Fig. 4: Confusion Matrix for the ANN Model

E. Training Dynamics and Convergence Behavior Analysis

Figure 5 depicts accuracy and loss trajectories that convincingly demonstrate the model's steady improvement over several training epochs. The accuracy curve, as shown on the left, exhibits a sharp and consistent increase in the early stages, followed by a gradual convergence into nearly flawless categorization around epoch 40. Remarkably, the validation accuracy consistently tracks above the training curve, a desirable trait indicating that the model is not only learning effectively but also generalizing well to unseen data without signs of overfitting. This behavior underscores the success of the model's regularization strategy, particularly the integrated dropout and early stopping mechanisms.

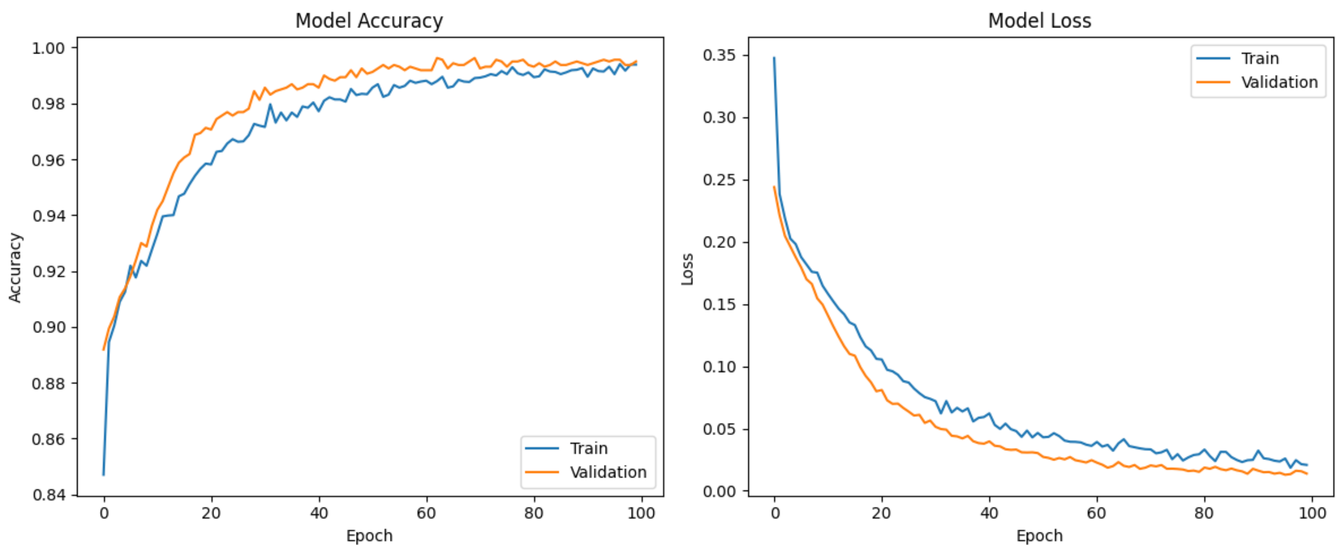

Fig. 5: ANN Model’s loss and accuracy

The matching loss curves on the proper support show this stability. As the model adjusts its internal parameters, the training loss tapers off from a steep initial decline. The validation loss, which is noticeably reduced throughout the training cycle, indicates that the model is capturing a significant structure in the data rather than merely fitting noise or outliers. A mature state of convergence is indicated by the smooth, parallel flattening of both loss curves around epoch 100, at which point additional training produces marginal gains. These curves show a well-calibrated learning process that is effective, steady, and robust, demonstrating the validity of the optimization regimen and the model architecture.

F. Feature Importance Analysis Based on Permutation Score

The bar plot in Figure 6 shows the permutation-based feature importance analysis, which shows the extent to which each input variable influences the model's ability to predict. With a relevance value of around 0.0139, CGPA stands out as the most critical factor influencing student placement results. With a score of 0.0128, Projects_Completed comes in second, suggesting that practical project experience influences placement prospects almost as much. With a noteworthy contribution of 0.0077, Communication Skills comes in third place, highlighting the importance of verbal and interpersonal skills in terms of employability. The effect of current academic trends is shown in Prev_Sem_Result's mild influence score of 0.0066. Nevertheless, IQ makes a negligible contribution (0.0025), indicating that applied academic success is more important than innate cognitive ability, even though the latter is crucial. Conversely, the model's predicted accuracy is not significantly impacted by Academic Performance ($p = 0.0011$), Internship Experience ($p = 0.0008$), or Extracurricular Score (negligible). Together, these results underscore the importance of long-term academic consistency, real-world application through projects, and practical communication skills in making placement decisions. They also show that short-term performance metrics and extracurricular activities are relatively insignificant in this regard.

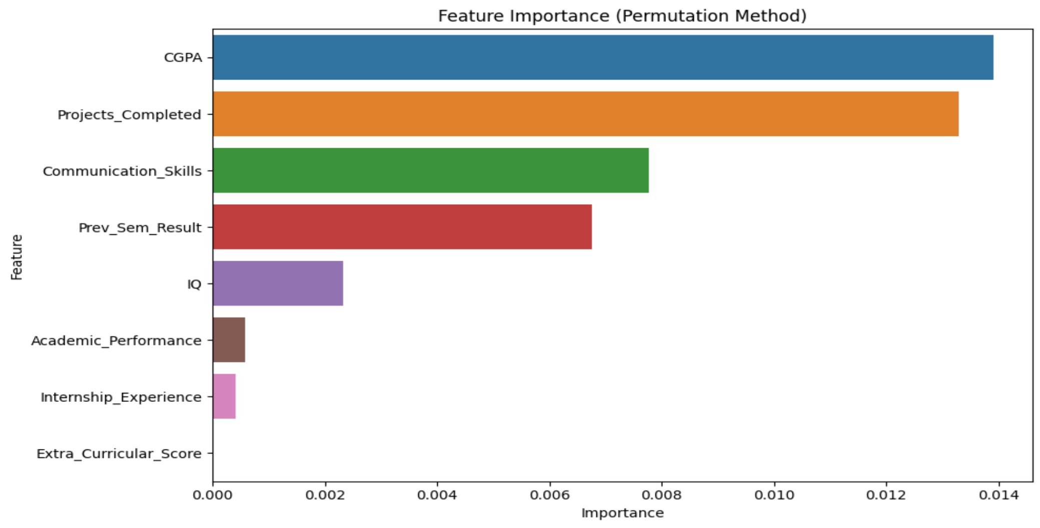

Fig. 6: Feature importance

V. CONCLUSION AND FUTURE WORK

Understanding the factors that influence student placement is becoming increasingly important, primarily as educational institutions aim to support students in preparing for competitive job markets. In this study, we examined the performance of various machine learning techniques—ranging from basic classifiers to deep learning models—when applied to student data that includes academic records, internship history, and aptitude-related metrics. The idea behind this study came from a simple concern: students want to know where they stand when it comes to placement, and institutions want better ways to guide them. We looked into how machine learning models could help predict placement outcomes, based on academic records and a few other features. We ran several models and compared their results. The ANN came out on top — not because it was perfect, but because it seemed better at picking up on weird, non-linear stuff in the data that simpler models missed. Logistic Regression and KNN didn't hold up well, and while Random Forest was decent, it felt like it was trying too hard — probably overfitting in some spots.

But we're still far from a complete solution. The dataset could be improved. We only looked at a limited number of features. If we had more info — like what kind of extracurriculars students were involved in, or what kind of real-world experience they had — the model could probably do more. Also, trying this on data from other institutions would help test if the model is just memorizing patterns or generalizing. Another thing worth mentioning is that most ML models are still a bit of a black box. If future work could focus on making them easier to interpret, it'd be way more useful for teachers or counselors who need to explain to students *why* they're at risk or what they can do better.

REFERENCES

1. Li, Q., Swaminathan, H., & Tang, J. (2009). Development of a classification system for engineering student characteristics affecting college enrollment and retention. *Journal of Engineering Education, 98*(4), 361–376.
2. Selingo, J. J. (2013). *College (un)bound: The future of higher education and what it means for students*. Houghton Mifflin Harcourt.
3. Armstrong, W. B. (2000). The association among student success in courses, placement test scores, student background data, and instructor grading practices. *Community College Journal of Research & Practice, 24*(8), 681–695.
4. Naseer, F., Tariq, R., Alshahrani, H. M., Alruwais, N., & Al-Wesabi, F. N. (2025). Project based learning framework integrating industry collaboration to enhance student future readiness in higher education. *Scientific Reports, 15*(1), 24985.
5. Tazehkand, S. A. (2024). Enhancing student graduation rates by mitigating failure, dropout, and withdrawal in introduction to statistical courses using statistical and machine learning.
6. 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. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1909–1918).
7. Maurya, L. S., Hussain, M. S., & Singh, S. (2021). Developing classifiers through machine learning algorithms for student placement prediction based on academic performance. *Applied Artificial Intelligence, 35*(6), 403–420.
8. Çakıt, E., & Dağdeviren, M. (2022). Predicting the percentage of student placement: A comparative study of machine learning algorithms. *Education and Information Technologies, 27*(1), 997–1022.
9. Maurya, L. S., Hussain, M. S., & Singh, S. (2022). Machine learning classification models for student placement prediction based on skills. *International Journal of Artificial Intelligence and Soft Computing, 7*(3), 194–207.
10. Maragatham, T., Yuvarani, P., Harishri, S., & Swetha, R. (2024). Student placement prediction using deep learning techniques. In *2024 8th International Conference on Electronics, Communication and Aerospace Technology (ICECA)* (pp. 620–626). IEEE.
11. Kadu, R., Assudani, P. J., Mukewar, T., Kapgate, J., & Bijekar, R. (2024). Student placement prediction and skill recommendation system using machine learning algorithms. In *2024 International Conference on Inventive Computation Technologies (ICICT)* (pp. 401–408). IEEE.

12. Elias, A. H., Khairi, F. A., & Elias, A. H. (2025). Hybrid machine-learning framework for predicting student placement. *Journal of Transactions in Systems Engineering, 3*(2), 403–419.
13. Rao, K. E., Pydi, B. M., Vital, T. P., Naidu, P. A., Prasann, U., & Ravikumar, T. (2023). An advanced machine learning approach for student placement prediction and analysis. *International Journal of Performability Engineering, 19*(8), 536.
14. Hegde, V., Abhinav, M., & Roshin, C. (2023). Predicting student placement using PCA and machine learning technique. In *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)* (pp. 1–5). IEEE.
15. Fathima, A. S., Basha, S. M., Ahmed, S. T., Khan, S. B., Asiri, F., Basheer, S., & Shukla, M. (2025). Empowering consumer healthcare through sensor-rich devices using federated learning for secure resource recommendation. IEEE Transactions on Consumer Electronics.
16. Ahmed, S. T., Fathima, A. S., & Reema, S. (2023, December). An Improved System for Students Feedback Analysis Using Supervised Probability Techniques. In 2023 10th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON) (Vol. 10, pp. 328-333). IEEE.
17. Ahmed, S. T., Bhushan, S. B., Srinivas, A. S., Basha, S. M., & Reddy, B. (2024, February). Estimating the impact of engineering education among students in India using machine learning and deep learning techniques. In AIP Conference Proceedings (Vol. 2742, No. 1, p. 020111). AIP Publishing LLC.