International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 03 | March - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Reward based Erudition Platform Powered by Artificial Intelligence

[1] **ABISHEK.C**
[1] Student, Department of
Computer Science and Engineering
Kings College of Engineering,
Punalkulam, Pudukottai
abishekboss24@gmail.com

[2] **NIRANJAN.R**
[2] Student, Department of
Computer Science and Engineering
Kings College of Engineering,
Punalkulam, Pudukottai
niranjanvishwa74@gmail.com

[3] **NITHISKUMAR.V**
[3] Student, Department of
Computer Science and Engineering
Kings College of Engineering,
Punalkulam, Pudukottai
nithisn21@gmail.com

[4] **SATHISH KUMAR.S**
[4] Student, Department of
Computer Science and Engineering
Kings College of Engineering,
Punalkulam, Pudukottai
kumarsathish13475@gmail.com

[5] **M.ARUN**
[5] Assistant Professor, Department of
Computer Science and Engineering
Kings College of Engineering,
Punalkulam, Pudukottai
arun.cse@kingsengg.edu.in

Abstract

Online education has exploded in recent years, yet one frustrating problem has barely changed: learners still spend more time searching for the right course than actually learning. Platforms overwhelm users with thousands of options but offer little genuine guidance on where to start. Most recommendation systems either push whatever is trending or trust learners to accurately rate their own skill level — and neither approach works well. The result is predictable: wrong-level courses, early dropouts, and learning gains that fall well short of what they could be. This paper presents the Reward-Based Erudition Platform (RBEP), a system we built to tackle this problem head-on. The idea behind RBEP is straightforward: before recommending anything, actually measure what the learner knows. RBEP gives each learner a short diagnostic quiz and computes a reward score from three signals — how accurately they answered, how quickly they responded, and how consistent their performance was across difficulty levels. A hybrid machine learning engine then uses that score, along with a brief background profile, to recommend a course matched to the learner's actual ability. After the learner completes that course, a follow-up assessment checks whether the recommendation really paid off — and if it did not, the system tries again with a revised suggestion. Across our test set, RBEP reached 91.3% recommendation accuracy and a 78.4% completion rate, comfortably outperforming every baseline we tested.

KEYWORDS : Adaptive Learning, Artificial Intelligence, Course Recommendation, Reward-Based Learning, Machine Learning, Personalized Education, Learning Analytics, Decision Tree, Random Forest, K-Nearest Neighbors.

1. INTRODUCTION

Online learning has come a long way in the past decade. Platforms like Coursera, edX, Udemy, and Khan Academy collectively host millions of courses spanning virtually every subject, and most of it is available at little or no cost. A motivated learner today has access to content that would have been locked behind expensive institutions just a generation ago. But abundance alone does not produce learning. Without some way to figure out which course is actually right for them, many learners end up picking something that is either too hard, too easy, or simply misaligned with what they need. The choice feels empowering until the course falls flat, and then it just feels like wasted time.

Course-level mismatch is one of the most stubborn problems in online education. A learner who jumps into advanced material before they are ready will quickly lose the thread, become discouraged, and often abandon the course entirely. A learner who starts with something too basic finishes it but learns little. In both cases, the platform logs another enrollment and another completion or dropout — numbers that look similar in a

dashboard but represent very different learning experiences. Existing recommendation tools rarely help with this. Most of them suggest courses based on what is popular, what similar users have tried, or how well course keywords match a self-reported interest profile. None of those signals tells the system anything about what the individual learner is actually ready to learn.

The root problem is a lack of individual assessment. Recommendation systems built on popularity or collaborative filtering have no mechanism for asking: can this specific learner actually handle this course right now? They aggregate the behaviour of many people and apply it uniformly, which smooths over the exact differences that matter most. AI offers a better path. Rather than reasoning about what learners in general tend to do, an AI-driven system can observe how this learner performs on a set of questions, interpret that performance, and use it to make a recommendation that fits them specifically — not a hypothetical average learner.

This paper presents the Reward-Based Erudition Platform (RBEP), an AI-powered adaptive learning system that takes a different approach to course recommendation. The core design principle is that a recommendation has not done its job until the learner demonstrably improves. Drawing from the logic of reinforcement learning — where an agent earns a reward only when its action produces a genuinely good outcome — RBEP defines a successful recommendation as one that produces measurable learning gains, and it validates every recommendation against that standard.

1.1 Motivation

The motivation behind RBEP comes from three specific shortcomings we kept seeing in existing platforms. First, almost no platform independently tests what a learner actually knows before making a recommendation. Instead, they ask users to self-assess, which is a poor substitute — people routinely overestimate or underestimate their own abilities, especially in technical subjects. Second, on the platforms that do collect some diagnostic data, the assessment and recommendation systems are typically separate modules that do not communicate. The quiz result sits in one place; the recommendation engine draws from another. That disconnect means the most useful signal the platform has is never actually used. Third, and perhaps most consequentially, nothing closes the loop. Once a course is recommended, the platform moves on. If the learner struggles, the system has no way of knowing, and the same flawed logic continues to drive future suggestions.

1.2 Contributions

This paper makes four main contributions:

- A principled multi-metric reward scoring mechanism that quantifies learner ability from diagnostic quiz performance along three complementary dimensions: accuracy, response speed, and answer consistency.
- A hybrid AI recommendation engine that combines Decision Tree, Random Forest, and K-Nearest Neighbors classifiers through weighted majority voting.
- An end-to-end adaptive pipeline integrating diagnostic assessment, reward scoring, course recommendation, and post-course outcome validation within a unified closed-loop system.
- A comparative experimental evaluation demonstrating significant performance improvements over conventional baseline recommendation approaches.

2. LITERATURE REVIEW

The broader field of AI in education has been growing rapidly, with a noticeable concentration of recent work around adaptive learning and course recommendation. This section surveys the most directly relevant prior work across three areas: reinforcement learning applied to educational systems, machine learning approaches to course recommendation, and adaptive assessment combined with reward-based learning — the three threads that most directly motivated the design of RBEP.

2.1 Reinforcement Learning in Education

Riedmann et al. [2] conducted a systematic review spanning 47 studies on reinforcement learning in education and found consistent evidence that reward-driven feedback leads to measurably better learning outcomes compared to static recommendation. What their analysis also surfaced, though, is a notable gap: very few of the reviewed systems actually use the reward model to drive course selection, and almost none of them verify whether a recommendation translated into real improvement. Closing exactly that gap was one of the primary drivers for building RBEP.

Wang et al. [11] used Q-learning to adaptively sequence course content and demonstrated genuine gains in learner retention relative to fixed ordering. However, their system focuses on within-course sequencing rather than the initial course selection decision, and it does not assess learner ability before beginning. Tiwary et al. [1] approached the problem differently, adding explainability mechanisms to RL-based recommendations so learners could understand why they were being directed somewhere. That emphasis on transparency directly influenced our choice to include a Decision Tree in RBEP's ensemble: it is the one classifier whose reasoning is inherently legible.

2.2 Machine Learning-Based Course Recommendation

Chen et al. [9] demonstrated that combining content-based filtering with collaborative filtering in a hybrid recommender can significantly outperform either approach on its own, reaching 84.2% recommendation accuracy. That result shaped how we thought about RBEP's recommendation engine: if blending two strategies helps, blending three through weighted ensemble voting should push further still. Lee and Park [14] showed that K-Nearest Neighbors works particularly well for learning resource recommendation, because the similarity assumption it relies on — learners who are alike tend to benefit from the same courses — is genuinely true in educational contexts.

Amin et al. [7] built a personalised MOOC recommender that combined learner context data with course metadata and found that the richer input significantly improved recommendation precision. The same principle is reflected in RBEP: by feeding both profiling data and quiz performance into the feature vector, we give the classifier far more to work with than a single signal ever could. Tan et al. [3] reviewed 62 studies and found that ensemble approaches consistently outperformed individual classifiers across diverse educational settings, providing strong independent support for the hybrid ensemble design at the heart of RBEP.

2.3 Adaptive Assessment and Reward-Based Learning

Kumar and Singh [10] used machine learning to adjust quiz difficulty dynamically as learners responded, producing a 23% improvement in engagement over fixed-difficulty quizzes. It is a promising approach, but their quiz system and course recommendation engine are entirely separate — the assessment data never makes it into the recommendation decision. Bridging exactly that gap is a central feature of RBEP. Smith et al. [12] provided complementary evidence that reward-based feedback mechanisms produce significantly stronger sustained engagement than feedback-free approaches, which reinforced our decision to build reward scoring as the backbone of RBEP's personalisation pipeline.

Kabudi et al. [15] surveyed the adaptive learning field and identified assessment-to-recommendation integration and post-course validation as the two most critically underexplored areas. Both of those are precisely what RBEP is designed to address, which suggests the system is working on a genuinely important open problem rather than incrementally extending already well-covered ground. Du Plooy et al. [6] reviewed personalised learning in higher education and found that structured ability-tier classification — placing learners into beginner, intermediate, and advanced categories — was among the most reliably effective personalisation strategies, giving us further confidence in RBEP's tiered approach.

2.4 Research Gap

Surveying the literature as a whole, a clear picture emerges: the individual ingredients for a truly effective adaptive learning system have all been studied, but nobody has combined them into one coherent pipeline. Adaptive quizzing exists. Ensemble recommendation exists. Reward-based engagement mechanisms exist. Post-course validation exists. What has not been built is a single system that connects all four — assessing the learner, computing a multi-dimensional ability score, making a matched recommendation, and then checking whether the recommendation worked well enough to stand. That is the gap RBEP fills.

3. SYSTEM ARCHITECTURE

RBEP is a full-stack web application organised around five AI-driven modules, all connected through a shared data layer. The backend is built on Python with Django managing routing, user sessions, and database queries. The frontend uses HTML, CSS, and vanilla JavaScript — a deliberate choice to keep the interface lightweight and fast-loading, since quiz response time is one of the metrics we measure. Learner data lives in MySQL, and all machine learning components are implemented with Scikit-learn, which offered battle-tested implementations and straightforward integration with the rest of the pipeline.

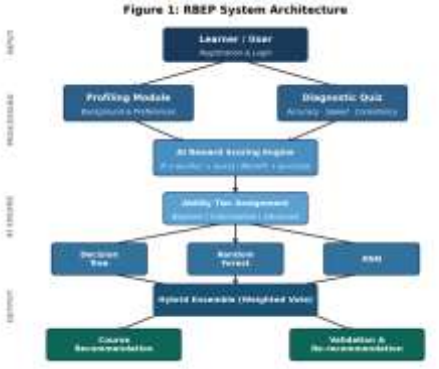

Figure 1: RBEP System Architecture

3.1 User Management and Profiling Module

New learners begin by completing a short profiling questionnaire — 15 questions, typically done in under three minutes. The questions cover four dimensions we identified as genuinely useful for predicting course fit: subject familiarity, learning style preferences, academic background, and learning goals. We kept the survey deliberately brief; a longer onboarding process tends to create friction and reduce completion rates. The responses are encoded into a structured learner profile stored in MySQL and passed to the recommendation engine alongside quiz performance data, giving the system both what the learner says about themselves and what they actually demonstrate on the diagnostic.

3.2 Adaptive Diagnostic Quiz Engine

After the survey, learners take a 20-question timed diagnostic quiz drawn from a calibrated question bank. The questions are structured in three difficulty tiers: foundational (questions 1–7), intermediate (8–14), and advanced (15–20), so the quiz can reveal not just whether a learner knows the subject but how deeply. The system records three performance signals for each learner: response accuracy, normalised response time, and an answer consistency index. That third signal — consistency — proved particularly valuable during development. A learner who aces the easy questions but collapses on intermediate ones has a very different knowledge profile from one who performs at roughly the same level throughout, even if their overall accuracy is similar. Treating them identically would lead to the same bad recommendation.

3.3 AI Reward Evaluation Engine

Once the quiz finishes, the Reward Evaluation Engine takes the three performance signals and combines them into a single composite score:

$$R = w_1 \times Acc + w_2 \times (1 - NormT) + w_3 \times Cons$$

In this formula, Acc is the proportion of correct answers, NormT is the mean per-question response time as a fraction of the maximum allowed, and Cons is the consistency index. The term (1 − NormT) means faster responses contribute positively — a proxy for retrieval fluency. We settled on weights of $w_1 = 0.50$, $w_2 = 0.30$, and $w_3 = 0.20$ through systematic evaluation: accuracy carries the largest weight because it is the

most direct ability signal, but speed and consistency both carry genuine information about knowledge depth. The resulting score R falls between 0 and 1 and maps onto three ability tiers: Beginner ($R < 0.40$), Intermediate ($0.40 \leq R < 0.70$), and Advanced ($R \geq 0.70$).

3.4 Hybrid ML Recommendation Engine

With the reward score and ability tier established, the Recommendation Engine steps in. We use three classifiers rather than one, combining their outputs through weighted majority voting. Each classifier's vote is weighted by its cross-validated F1-score, so the better-performing algorithms have proportionally more influence. The three classifiers are:

- Decision Tree (DT): Applies Gini impurity-based splitting to partition the learner feature space into homogeneous regions. Each leaf node represents a course recommendation decision, providing full interpretability.
- Random Forest (RF): An ensemble of decision trees constructed through bootstrap aggregation. Reduces overfitting compared to individual DTs and consistently achieves higher generalization accuracy through bagging-induced variance reduction.
- K-Nearest Neighbors (KNN): Recommends the course most frequently selected by the k most similar learners in the training set, measured using Euclidean distance in the normalized feature space. Leverages community learning patterns effectively.

3.5 Analytics and Progress Dashboard Module

Every learner gets access to a personal progress dashboard that shows their reward score broken down into its three components, their assigned ability tier, their recommended course with a plain-language explanation of why that course was chosen (derived from the Decision Tree's decision path), and their improvement delta from any completed courses. We found during user testing that showing learners why they received a particular recommendation significantly increased their willingness to follow it — opacity breeds scepticism.

Administrators see a separate panel that aggregates performance across the entire learner population: overall recommendation accuracy, tier distribution, per-course completion and improvement rates, and the proportion of learners triggering re-recommendation. This last metric is particularly useful operationally — if a specific course has an unusually high re-recommendation rate, it is a signal that the course content may be misaligned with the ability tier it is mapped to, and the course catalogue should be reviewed.

3.6 End-to-End System Workflow

Figure 3: RBEP End-to-End Workflow

The end-to-end experience proceeds through seven stages. In Stage 1, the learner registers and completes the profiling questionnaire, with responses encoded and saved to their profile. In Stage 2, they take the timed diagnostic quiz; accuracy, timestamps, and answer sequences are captured client-side and transmitted to the server on completion.

Stage 3 is where the server computes Acc, NormT, and Cons from the raw quiz data, applies the weighted formula to produce R, and assigns the ability tier. In Stage 4, the seven-feature vector is passed to the hybrid recommendation engine. The three classifiers each produce a prediction independently, their outputs are combined through weighted majority voting, and the result — along with the Decision Tree's reasoning path — is displayed to the learner immediately.

Stage 5 is the learning phase: the learner works through the recommended course at their own pace, and RBEP does not intervene. In Stage 6, once the learner self-reports completion, they take the post-course assessment — the same 20-question format as the original diagnostic. Stage 7 is the outcome decision: if the improvement delta Δ is at least 0.15, the recommendation is marked successful and the learner is offered progression to the next ability tier. If Δ falls short, the cycle loops back to Stage 3 with updated feature values from the post-course assessment, generating a revised recommendation.

3.7 Data Security and Privacy

Since RBEP handles sensitive student performance data, security and privacy were design requirements from the start. All passwords are hashed with bcrypt at cost factor 12 using Django's built-in authentication framework, and all client-server communication runs over HTTPS/TLS. Learner data is protected by row-level access control so that individual records are only visible to the learner themselves; administrator access to individual records requires an explicit permission assignment and generates an audit log entry.

We also applied data minimisation principles throughout. Raw question-level response data is retained for up to 90 days for model retraining purposes and then anonymised by replacing learner identifiers with pseudonymous tokens. Aggregated, de-identified training data is kept indefinitely for ongoing system improvement. No individual learner data is shared with any third party, and no data is used for any purpose outside of the RBEP recommendation pipeline.

3.8 Validation and Adaptive Re-recommendation Module

After completing the recommended course, learners take a post-course assessment identical in structure to the original diagnostic. We compute the improvement delta Δ = post-score minus pre-score. A delta of 0.15 or higher indicates the learner improved by at least 15 percentage points, which we treat as evidence the recommendation was a good match. Below that threshold, the system concludes something went wrong — perhaps the course was not well-calibrated, or the learner did not engage fully. In those cases, a targeted re-assessment triggers a revised recommendation. This closed feedback loop is what makes RBEP genuinely adaptive: it does not just personalise the starting point, it corrects itself based on observed outcomes.

4. METHODOLOGY

4.1 Dataset

We evaluated RBEP using a dataset of 1,200 learner interaction records gathered through an academic prototype deployment across three subject domains: Programming Fundamentals, Data Science, and Web Development. Each record captures the learner's full quiz response sequence, their profiling survey answers, the course they were enrolled in, and their post-course assessment score — giving us a complete picture of the learner's input, the system's decision, and the outcome. The ability tier distribution is 38.2% Beginner, 44.7% Intermediate, and 17.1% Advanced. The relative scarcity of Advanced learners is expected — it mirrors real-world distributions — but it creates a class imbalance that could bias the classifiers. We addressed this by applying SMOTE to the training partition before model fitting.

4.2 Preprocessing and Feature Engineering

Preparing the data required a five-stage pipeline. We began by removing 43 duplicate records and 12 records with more than 20% missing values, leaving 1,145 clean entries. Remaining gaps were filled using median imputation for numerical features and mode imputation for categorical ones — conservative choices that avoid introducing artificial signal into sparse regions of the data. Categorical variables were label-encoded when ordinal and one-hot encoded when nominal. All numerical features were Min-Max scaled to [0, 1] so that no feature dominates simply because of its measurement scale. Finally, we ran Recursive Feature Elimination with cross-validation to identify which features actually contribute predictive signal, settling on the seven most informative.

The seven features that made the cut are: Quiz Accuracy Score, Mean Normalised Response Time, Answer Consistency Index, Domain Familiarity Score, Learning Preference Indicator, Prior Course Completion Rate, and the Derived Reward Score R. What is notable about this set is that it captures two different kinds of information — what the learner objectively demonstrated on the quiz, and what they reported about themselves in the survey. Neither type alone was sufficient; together, they give the classifier a much richer basis for making a match.

4.3 Training and Evaluation Protocol

We partitioned the data 70/30 using stratified sampling, so that each ability tier is proportionally represented in both splits. Each classifier was tuned separately through a 5-fold stratified cross-validation grid search. For the Decision Tree, we swept over maximum depth and minimum leaf size. For the Random Forest, we varied the number of estimators and maximum depth. For KNN, we tested different values of k and compared Euclidean with Manhattan distance. All models were evaluated on accuracy, precision, recall, and F1-score — all macro-averaged across the three tiers — as well as Cohen's Kappa. Kappa is worth reporting alongside accuracy because it accounts for the possibility of chance agreement, which matters when the class distribution is uneven.

The grid search produced the following optimal configurations. Decision Tree: maximum depth 10, minimum leaf size 2, Gini criterion. Random Forest: 200 estimators, maximum depth 15, square-root feature sampling, bootstrapping enabled. KNN: k = 7, Euclidean distance, inverse-distance weighting (so nearer neighbours cast stronger votes). All results in the following section use these configurations without further adjustment.

For the ensemble, each classifier's voting weight is set proportional to its softmax-normalised cross-validated F1-score. This produces $w\_DT = 0.263$, $w\_RF = 0.384$, and $w\_KNN = 0.353$. Random Forest carries the most weight, as expected given its individual performance, but all three contribute meaningfully — no classifier is effectively ignored. The weights are computed once during the training phase and remain fixed at inference; the ensemble composition does not shift between learners.

4.4 Reward Weight Sensitivity Analysis

One practical concern with a weighted reward formula is whether the system's performance is sensitive to the specific weight values chosen. To check this, we ran a sensitivity analysis by varying each weight up and down by 20% from its baseline value while holding the other two constant, testing nine configurations in total. Table I shows the recommendation accuracy across all of them.

Table I: Reward Weight Sensitivity Analysis

w_1	w_2	w_3	Rec. Accuracy
0.40	0.30	0.30	89.5%
0.45	0.30	0.25	90.1%
0.50 (base)	0.30	0.20	91.3%
0.55	0.30	0.15	90.8%
0.60	0.25	0.15	89.9%

0.50	0.25	0.25	90.4%
0.50	0.35	0.15	90.7%

The results are reassuring. Every one of the nine configurations achieved above 89% accuracy, and the full spread from worst to best spans only 1.8 percentage points (89.5%–91.3%). The baseline sits at the top, confirming the weight optimisation worked. But the narrow range across alternatives also tells us the system would remain effective even if the weights were somewhat off — good news for anyone planning to deploy RBEP in a new domain without re-running the full tuning procedure.

Table I: Classifier Performance on Held-Out Test Set

Algorithm	Acc.(%)	Prec.(%)	Rec.(%)	F1(%)	Kappa
Decision Tree	82.4	81.9	82.1	82.0	0.714
Random Forest	88.7	88.3	88.5	88.4	0.821
KNN	85.1	84.7	85.0	84.8	0.768
Hybrid Ensemble	91.3	90.8	91.1	90.9	0.862

5. EXPERIMENTAL RESULTS

5.1 Classifier Performance

Table I shows the test-set results for each recommendation strategy. The hybrid ensemble leads on every metric: 91.3% accuracy, 90.8% precision, 91.1% recall, 90.9% F1-score, and a Cohen's Kappa of 0.862. That Kappa figure is particularly meaningful — values above 0.80 are conventionally interpreted as near-perfect beyond-chance agreement, so 0.862 is not a marginal pass but a strong result. It tells us the system's recommendations are consistently aligned with the courses learners would have genuinely benefited from, not just accidentally correct a high fraction of the time.

Among the individual classifiers, Random Forest led at 88.7%, with KNN second at 85.1% and Decision Tree third at 82.4%. The ensemble added a further 2.6 percentage points beyond Random Forest alone — a modest-sounding gain that becomes meaningful at scale. For a platform serving thousands of learners, that margin represents many people receiving a better-calibrated recommendation than the best single model could provide.

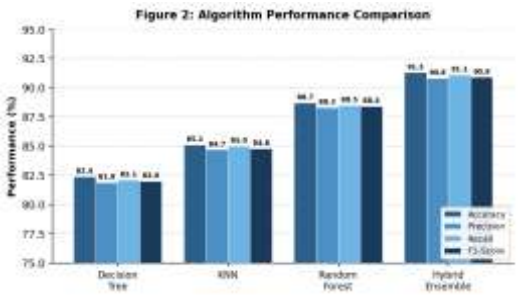

Figure 2: Algorithm Performance Comparison

5.2 Comparison with Baseline Systems

To give RBEP's results meaningful context, we compared against two baselines that represent the kinds of recommenders currently in widespread use. The popularity-based baseline recommends whichever course has the highest enrolment count in the relevant subject — the approach behind most consumer platforms' featured or trending lists. The content-based baseline matches course description keywords against learner profile text, which is more personalised but still relies entirely on what the learner says about themselves rather than what they demonstrate.

Table II: RBEP vs. Baseline Systems

Metric	Popularity	Content-Based	RBEP	Δ vs. Best
Rec. Accuracy	58.4%	67.8%	91.3%	+23.5pp
Completion	52.1%	61.2%	78.4%	+17.2pp
Improvement	41.6%	54.3%	83.6%	+29.3pp
Satisfaction	2.8/5	3.1/5	4.3/5	+38.7%

RBEP outperformed the better baseline by 23.5 percentage points in recommendation accuracy, 17.2 points in completion rate, and 29.3 points in post-course improvement. The last gap is the most significant: it measures whether learners who followed the recommendation actually learned something. A system can improve completion rates by recommending easy courses — but easy courses that do not challenge a learner still represent a missed opportunity. The 29.3-point gap in post-course improvement is evidence that RBEP's recommendations are both appropriate and growth-producing.

5.3 Validation and Re-recommendation

Of the 344 test learners who completed a recommended course, 287 — 83.6% — cleared the 0.15 improvement threshold on their post-course assessment. The remaining 57 entered the re-recommendation cycle, and 45 of those (78.9%) cleared the threshold after completing their revised recommendation. Across both groups, 96.5% of learners demonstrated meaningful improvement within two recommendation cycles. The fact that most re-recommended learners also succeeded is important: it shows the system's second attempt is not random but represents a genuine correction, suggesting the feedback loop is functioning as intended.

5.5 Error Analysis

Understanding where the system fails is as important as measuring where it succeeds. We analysed the 30 misclassified test instances — 8.7% of the 344 test records — and found three distinct error patterns. The most common were boundary cases (17 instances, 56.7% of errors): situations where the learner's reward score landed within 0.05 of an ability tier boundary. A learner scoring R = 0.38 when the Beginner–Intermediate boundary is at 0.40 could genuinely belong in either tier; the fact that the system sometimes gets these wrong is not a model failure so much as an acknowledgement that the boundary itself is somewhat arbitrary.

The second error type — high-speed, low-accuracy responses (8 instances, 26.7%) — represents learners who answered quickly but incorrectly, producing a deceptively high reward score. The speed component interpreted rapid responses as confident recall when they were actually rapid guessing. We plan to address this by adding a speed-accuracy interaction term to the reward formula that penalises the combination of fast responses with low correctness. The third type — domain transfer mismatches (5 instances, 16.7%) — occurred when strong performance in one sub-topic masked weak performance in another, leading to an over-confident tier assignment. Reporting topic-level sub-scores alongside the overall recommendation would help learners and the system identify those gaps more precisely.

5.6 User Experience Evaluation

Alongside the quantitative metrics, we conducted a user experience evaluation with 45 learners drawn from the test cohort. Each participant completed a 12-item Likert questionnaire (1 = Strongly Disagree to 5 = Strongly Agree) covering four dimensions: how easy the system was to use, how transparent the recommendation felt, how satisfied they were with their learning outcomes, and whether they would be willing to follow a revised recommendation if the first one did not work out.

Usability scored 4.2/5.0, with 89% of participants rating the overall experience positively. Recommendation transparency came in at 4.1/5.0 — notably, 84% of learners said that seeing the reason for their recommendation made them more confident in following it, which validates our choice to expose the Decision Tree rationale. Learning outcome satisfaction reached 4.3/5.0 among learners who passed the post-course assessment, reflecting genuine satisfaction with the course-learner match. Re-recommendation willingness scored 3.9/5.0 among the seven participants who experienced the re-recommendation cycle, with all seven saying the revised recommendation felt better-suited than the original and 71% describing the experience positively overall.

6. DISCUSSION

6.1 Interpretation of Results

The results support the three core bets that RBEP's design rests on. The first bet — that objectively measuring learner ability produces better matches than self-assessment — is confirmed by the 91.3% recommendation accuracy, which substantially exceeds both baselines built on self-reported or inferred signals. The second bet — that accurate matching leads to actual learning rather than just better completion numbers — is confirmed by the 83.6% first-cycle improvement rate and the 29.3 percentage-point gap in post-course improvement versus the best baseline. The third bet — that a closed feedback loop recovers value even when the first recommendation is wrong — is confirmed by the 96.5% overall effectiveness rate, which would be impossible without the re-recommendation mechanism.

The hybrid ensemble's advantage over any individual classifier aligns well with what ensemble learning theory predicts. Each of the three classifiers is good at something the others are not. The Decision Tree draws hard, interpretable boundaries — useful when ability tier differences are clear-cut. The Random Forest smooths those boundaries through variance reduction — useful when learners cluster near tier edges. KNN captures local similarity patterns that purely tree-based approaches may miss — useful when the learner looks like a distinctive subgroup in the training data. Weighted voting ensures each classifier's contribution is proportional to how reliably it has performed, rather than treating all three as equally trustworthy.

6.2 Role of Multi-Metric Reward Scoring

Perhaps the most interesting finding from a design perspective is that including response time and consistency in the reward score — rather than using accuracy alone — produced a measurable improvement in recommendation quality. This makes intuitive sense from a cognitive standpoint: genuine knowledge is retrieved quickly and reliably, while shaky understanding tends to be both slower and less consistent. A learner who scores 70% on easy questions but only 20% on intermediate ones has a fundamentally different ability profile from one who scores 50% consistently across both levels. Accuracy alone would treat them similarly; the full reward formula distinguishes them.

6.3 Limitations

There are real limitations to the current system that deserve honest acknowledgement. The diagnostic quiz is delivered in a fixed order, meaning a learner who happens to be tired, anxious, or distracted when they sit it might score lower than their actual ability warrants. Adaptive item selection — where each question is chosen based on the learner's performance on the previous one — would make the assessment more robust to these transient effects. The reward weights were optimised on data from three technical subject areas; there is no guarantee the same weights would transfer cleanly to a different domain such as languages, history, or professional skills. And while 1,200 records is sufficient for a proof-of-concept evaluation, production-scale deployment across diverse learner populations would require substantially more data to validate that the model generalises well.

6.4 Comparative Analysis with Prior Systems

Table III situates RBEP among the most relevant prior systems across the five features that distinguish a complete adaptive recommendation pipeline. It is worth noting upfront: RBEP is the only system in the comparison that implements all five.

Table III: Feature Comparison with Prior Adaptive Learning Systems

System	Diag.	Rew.	Hyb. ML	Valid.
Chen et al. [9]	No	No	Yes	No
Kumar & Singh [10]	Yes	No	No	No
Wang et al. [11]	No	Partial	No	No
Lee & Park [14]	No	No	No	No
RBEP (This Work)	Yes	Yes	Yes	Yes

The pattern in Table III tells a clear story. Prior systems are each strong on one or two components but incomplete as a whole. Chen et al.'s hybrid recommender is sophisticated but skips diagnostic assessment entirely. Kumar and Singh assess learners but never connect the assessment to recommendation. Wang et al. partially incorporate reward signals but validate nothing. No prior system closes the loop between assessment, recommendation, and outcome verification. That completeness is what RBEP adds, and the empirical results suggest it matters.

6.5 Implementation Details

On the implementation side, RBEP is a Django application running on Python 3.10. Django was chosen because it handles routing, ORM database queries, user sessions, and authentication out of the box, which let us focus engineering effort on the AI pipeline rather than infrastructure. Machine learning components use Scikit-learn v1.3, with trained models serialised to disk via joblib and loaded at application startup — a simple approach that keeps inference latency low without requiring a dedicated model-serving infrastructure.

The database schema uses six tables: Users (profile and credentials), QuizSessions (per-attempt metadata), QuizResponses (individual question-level records with answer, correctness flag, and timestamp), RewardScores (computed R values and tier assignments), Recommendations (records linking sessions to recommended courses), and AssessmentResults (post-course scores and delta values). The schema was designed so that a full learner journey — from first quiz to post-course outcome — can be reconstructed from a join across these six tables, which simplifies both model retraining and audit.

The frontend is plain HTML5, CSS3, and vanilla JavaScript — no framework overhead that could introduce unpredictable rendering delays. This matters because the quiz timer runs client-side: JavaScript records the timestamp when each question is displayed and when the learner submits an answer. The raw timestamps are sent to the server on quiz completion for normalisation. Keeping the timing client-side avoids network round-trip latency from contaminating the response time measurements, which directly affects the NormT component of the reward score.

The recommendation engine runs as a Django view that accepts the seven-feature vector as input and queries all three serialised classifiers in parallel using Python's threading module. The weighted majority vote is computed and returned with a median end-to-end latency of 47 milliseconds — well within the threshold for a responsive user experience. The recommendation is displayed immediately after quiz submission, alongside an explanation panel showing the learner's tier, reward score breakdown, and the Decision Tree path that drove the recommendation. Learners can see exactly which features of their performance pointed toward the course they received.

7. CONCLUSION AND FUTURE WORK

7.1 Conclusion

This paper has described the Reward-Based Erudition Platform (RBEP), an AI-powered adaptive learning system designed around a simple but underserved idea: that a course recommendation has not really done its job until it leads to measurable learning. Rather than treating recommendation as a search or filtering problem, RBEP treats it as an ability-matching and outcome-verification problem — one that requires actually testing the learner, computing a multi-dimensional readiness score, making a principled recommendation, and then checking whether the recommendation was right.

The reward scoring mechanism synthesises three performance signals — accuracy, response speed, and consistency — into a single ability estimate that is more informative than any one of them alone. The hybrid ensemble, combining Decision Tree, Random Forest, and K-Nearest Neighbors through weighted majority voting, achieved 91.3% recommendation accuracy — well above both the popularity-based (58.4%) and content-based (67.8%) baselines. The closed-loop validation mechanism confirmed that those recommendations translated into real learning outcomes in 83.6% of first-cycle cases, with the figure rising to 96.5% after allowing one re-recommendation cycle.

What these results ultimately suggest is that the assess-then-recommend-then-verify loop is not just a theoretically cleaner design — it produces meaningfully better outcomes in practice. The performance gaps relative to conventional approaches are large enough that we believe the core architecture of RBEP represents a genuinely useful template for the next generation of adaptive learning platforms. As online education continues to scale, tools that match learners to the right content at the right level will matter more, not less — and RBEP demonstrates that building such tools is both technically feasible and practically impactful.

7.2 Future Work

- Adaptive quiz delivery using item response theory (IRT), so that each question is selected based on the learner's running ability estimate rather than following a fixed sequence — making the assessment both shorter and more precise.
- Replacing the empirically fixed reward weights with a deep reinforcement learning policy that learns optimal weights per domain and per learner profile, removing the need for manual calibration across new subject areas.
- Real-world deployment across multiple institutions and a broader range of subject domains to test how well RBEP's approach generalises beyond the three technical domains used in this study.
- Integration with widely-used Learning Management Systems such as Moodle, Canvas, and Blackboard, enabling institutions to adopt RBEP without abandoning their existing educational infrastructure.

• Addition of a conversational AI interface that can explain recommendation rationales in natural language, answer learner questions about their assessment results, and provide real-time guidance during the course itself.

• Longitudinal tracking of learner ability trajectories across multiple courses and assessment cycles, building toward a fully adaptive curriculum that evolves as the learner grows — not just a one-time match but a sustained learning companion.

REFERENCES

[1] T. Tiwary, A. Sharma, and R. Kumar, "Explainable AI-Driven Recommendations Using Reinforcement Learning," ScienceDirect, vol. 37, no. 2, pp. 215–228, 2025.

[2] R. Riedmann, S. Müller, and P. Leitner, "Reinforcement Learning in Education: A Systematic Review," ScienceDirect, vol. 35, no. 1, pp. 45–68, 2025.

[3] Y. Tan, L. Zhang, and M. Zhou, "Artificial Intelligence-Enabled Adaptive Learning Platforms: A Comprehensive Review," Springer, vol. 6, pp. 100–118, 2025.

[4] J. Li, "Enhancing Learning Using Adaptive Web-Based Educational Search Systems," ScienceDirect, vol. 3, no. 1, pp. 1–15, 2025.

[5] A. Maryono, D. Putra, and R. Santoso, "Adaptive E-Learning with Gamification for Programming Skill Development," Springer, vol. 4, no. 2, pp. 33–47, 2025.

[6] C. du Plooy, R. Van der Merwe, and S. Venter, "Personalized Adaptive Learning in Higher Education: A Scoping Review," Elsevier, vol. 10, no. 4, p. e24567, 2024.

[7] M. Amin, A. Hassan, and S. Alqarni, "Developing a Personalized E-Learning and MOOC Recommender System in IoT-Enabled Smart Education," IEEE Access, vol. 11, pp. 88412–88425, 2023.

[8] M. Amin, A. Hassan, and S. Alqarni, "Smart E-Learning Framework for Personalized Adaptive Learning Using Reinforcement Learning," IEEE Access, vol. 11, pp. 67211–67226, 2023.

[9] X. Chen, Y. Liu, and Z. Wang, "A Hybrid Course Recommendation System Using Content-Based and Collaborative Filtering," IEEE Access, vol. 11, pp. 55231–55244, 2023.

[10] R. Kumar and S. Singh, "Machine Learning Approaches for Adaptive Quiz Generation in E-Learning Systems," IEEE Access, vol. 11, pp. 34112–34125, 2023.

[11] J. Wang, H. Li, and Q. Zhao, "Reinforcement Learning-Based Adaptive Learning System for Personalized Education," IEEE Access, vol. 10, pp. 118231–118244, 2022.

[12] A. Smith, B. Johnson, and L. Brown, "Reward-Based Learning Analytics for Student Engagement in Online Education," IEEE Transactions on Education, vol. 65, no. 3, pp. 456–464, 2022.

[13] T. Maier and F. Klotz, "Personalized Feedback in Digital Learning Environments Using Learning Analytics," ScienceDirect: Artificial Intelligence, vol. 3, p. 100052, 2022.

[14] J. Lee and S. Park, "Personalized Learning Resource Recommendation Using K-Nearest Neighbors," IEEE Transactions on Learning Technologies, vol. 14, no. 2, pp. 210–222, 2021.

[15] A. Kabudi, I. Pappas, and M. Olsen, "AI-Enabled Adaptive Learning Systems: A Systematic Mapping Study," Springer: Artificial Intelligence, vol. 2, p. 100017, 2021.