Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 4, 2026

AN EXPLAINABLE CALIBRATION AND SHAP AUDIT OF HELP-SEEKING FEATURES IN CLASSICAL KNOWLEDGE TRACING

Dr. Mustafa Hameed[*1], **Dr. Musarat Karim**[2], **Dr. Muhammad Nauman**[3], **Ms. Alisha Fida**[4], **Dr. Nadia Khan**[5]

[*1, 2]*Department of Information Technology, Faculty of Computing, The Islamia University of Bahawalpur, Pakistan*
[3,4,5]*Department of Software Engineering, Faculty of Computing, The Islamia University of Bahawalpur, Pakistan*

DOI: https://doi.org/10.5281/zenodo.20506890

Keywords

Article History
Received: 11 March 2026
Accepted: 21 April 2026
Published: 12 April 2026

Corresponding Author: *
Dr. Mustafa Hameed

Abstract

Does richer feature engineering substitute for, add to, or simply lose out to model expressivity in classical knowledge tracing? We put this question to a direct test. In addition to the usual outcome-only baseline of prior accuracy and prior attempt counts, we layer two further families of behavioural features: a help-seeking block (rolling hint count, bottom-hint usage, attempts-per-problem, first-action rate) and a response-time block (rolling mean log-first-response-time and log-overlap-time). On the canonical ASSISTments 2009-10 Skill Builder corpus (433 161 attempts, 4 163 students, 123 skills), we fit PFA, LR-KT, LightGBM, and XGBoost across three nested feature blocks (outcome-only, +help-seeking, +response-time), reporting AUC, F1, ECE, and Brier with bootstrap 95 % CIs on a user-grouped 80/20 split. The headline result contradicts the prior literature. A well-tuned LightGBM on the six-feature outcome-only block was the single best configuration we found (AUC 0.838 [0.836, 0.841]; ECE 0.0099 [0.0081, 0.0123]); adding the help-seeking features actually lowered AUC to 0.832 and pushed ECE up to 0.0140, with CIs that did not overlap. TreeSHAP pins down the mechanism: the outcome features carry roughly 5× the global attribution magnitude of the help-seeking and response-time blocks combined, and the per-skill SHAP shows help-seeking doing its worst calibration damage precisely on the skills where the GBM is already miscalibrated. The linear LR-KT baseline behaves in the opposite direction, exhibiting a small ECE gain from the response time history at Block C. The pattern is consistent throughout: feature richness can stand in for model expressivity but does not compound with it, and on this corpus, expressivity wins. We released the pipeline, fitted models, per-skill reliability diagrams, and verified bibliography.

1. INTRODUCTION

Most published comparisons of classical knowledge tracing models on ASSISTments lean on a single metric, AUC on next-attempt correctness, and a single type of feature: counts and rolling accuracies derived from past *outcomes*. This convention has two blind spots. The first is behavioural. *Help-seeking* ; hint requests, bottom-hint use, and the choice of first action ; has been treated across the metacognitive tutor literature as a learning signal in its own right (Aleven et al. 2006; Roll et al. 2011; Nelson-Le Gall 1985), yet it seldom finds its way into the feature set of a classical KT model. The second concern is calibration, which the AUC does not measure. Two models can share an AUC and still differ sharply in how usable their probabilities are if one is over-confident in some skills and under-

confident in others. What the classical KT literature is missing, then, is an audit that moves *both* the feature set and the model class at once, and reports AUC, F1, ECE, and Brier with bootstrap CIs side by side.

We frame the study around three pre-registered research questions, all posed on the canonical ASSISTments 2009-10 Skill Builder release (Feng et al. 2009):

1. **RQ1.** *Does adding help-seeking history features (Block B) to an outcome-only KT baseline (Block A) improve next-attempt-correctness AUC, ECE, or Brier on a user-grouped held-out fold, and does the answer depend on the model class (PFA / LR-KT / LightGBM / XGBoost)?*
2. **RQ2.** *Does further adding response time history (Block C) improve any of the four metrics over Block B, and does the answer differ between the linear and tree-ensemble families?*
3. **RQ3.** *On a per-skill basis, do the help-seeking and response-time blocks improve calibration uniformly across the top-12 skills, or are there skills where the augmented blocks make calibration worse?*

Anticipating §5, the short answers run: for RQ1, no in the GBM family and only marginal for LR-KT; for RQ2, marginally positive for LR-KT but neutral-to-recovering for the GBM family; and for RQ3, per-skill calibration *degrades* on a small handful of skills (notably Add/Sub Integers, Mul/Div Integers, Angles), exactly those where the help-seeking signal is most confounded with the underlying skill difficulty. Therefore, this study presents a *negative result*: once a well-tuned GBM is in the pipeline, help-seeking and response-time features add little operational value. This has direct implications for course coordinators weighing whether to instrument hint logging. The mechanism behind this is that *model expressivity beats feature richness* in this corpus. A strong tree ensemble already wrings the available signal out of the outcome features alone, so the help-seeking layer only earns its keep with the linear baseline, which lacks expressivity to do the same.

No new estimator is offered here. Instead, we contribute a clean, fully explainable audit of when help-seeking signals help, when they hurt, and which model class actually profits from a richer feature set. The framing remains deliberately classical (Pelánek 2017): PFA and LR-KT fix the linear baseline, LightGBM and XGBoost capture whatever non-linear lift is available, and TreeSHAP traces each result back to the individual features. The code, fitted models, per-skill reliability diagrams, and verified bibliography accompany this paper.

2. Related Work

Classical knowledge tracing. Bayesian Knowledge Tracing (Corbett and Anderson 1995) is the canonical four-parameter HMM, with per-skill {L0, T, G, S}. Performance Factors Analysis (Pavlik et al. 2009) recasts BKT as a logistic regression on per-skill counts of prior success and failure, while Learning Factors Analysis (Cen et al. 2006) folds in a difficulty term and a per-student learning rate. Pelánek's broad comparison (Pelánek 2017) subsumes both under LR-KT, a logistic regression carrying the full skill × attempt cross-product. Knowledge Tracing Machines (Vie and Kashima 2019) go a step further, casting the whole family as factorisation machines and recovering BKT, PFA, and LFA as special cases. A complementary line individualises the parameters: Yudelson, Koedinger, and Gordon (Yudelson et al. 2013) provide BKT student-specific learn-rate and prior-knowledge terms and report that the individualised variant generalises better to unseen students. A later methodological audit by Pelánek (2018) makes a point we take seriously, namely that the evaluation protocol (cross-validation strategy, calibration metric, target horizon) tends to matter more than the choice of estimator within the classical family. We report the calibration next to AUC throughout.

Classical versus deep KT comparisons. Deep Knowledge Tracing (Piech et al. 2015) set off the LSTM-KT line, and two careful follow-ups then complicated the picture. Xiong et al. (Xiong et al. 2016) and Khajah, Lindsey & Mozer (Khajah et al. 2016) both show that DKT's headline lift over classical baselines on ASSISTments shrinks substantially once the data leakage in the original pipeline is removed, at which point well-tuned classical models recover most of the apparent deep net advantage. These architectures have since multiplied. Dynamic Key-Value Memory

Networks (Zhang et al. 2017) attach a key/value memory to track per-KC mastery, and self-attention variants such as SAKT (Pandey and Karypis 2019) and the encoder-decoder SAINT (Choi et al. 2020) post further AUC gains on the public benchmarks. Our study belongs to the sceptical branch of this debate: a strong classical baseline, paired with careful evaluation, tends to outperform naive comparisons that quietly conflate model class with evaluation rigor.

Help-seeking in intelligent tutoring. Aleven et al. (Aleven et al. 2006) introduce a normative model of help-seeking inside a cognitive tutor and put a name to the gap between *productive* hint use and *gaming-the-system* hint abuse. An earlier review by Aleven, Stahl, Schworm, Fischer, and Wallace (Aleven et al. 2003) sets up the broader framing: help-seeking is a metacognitive skill that *can be learned*. Roll, Aleven, McLaren, and Koedinger (Roll et al. 2011) back this up experimentally, showing that metacognitive feedback nudges students from abusive toward productive help-seeking. Aleven, Roll, McLaren, and Koedinger (Aleven et al. 2016) later consolidated a decade of cognitive tutor evidence into a qualified conclusion: on-demand help *does* support learning, but only when metacognitive scaffolding comes with it. This qualification is exactly why we treat help-seeking as its own feature category rather than a generic engagement proxy. The gaming-the-system pattern itself, repeated bottom-hint use without real engagement, is documented by Baker et al. (Baker et al. 2008), and it is what motivates carrying bottom_hint as a distinct feature in our Block B. Further back still, Nelson-Le Gall (Nelson-Le Gall 1985) supplies the developmental-psychology foundation, framing help-seeking as a self-regulatory learning skill. The line remains active: recent ITS work typifies distinct help-seeking profiles in mathematics tutoring (Melendez-Armenta et al. 2021), interactive learning studies revisit the achievement consequences of help-seeking versus help-abuse (Schulz and Voermanek 2025), and computing-education research keeps finding heterogeneous academic help-seeking across student groups (Ko 2023). This motivates the *features* rather than the model. A student who clicks straight through to the bottom hint on an unfamiliar skill is doing something behaviourally distinct from one who attempts the exercise twice before asking for help. Whether this distinction also predicts next-attempt correctness, the operational KT target, is an empirical question we set out to answer.

Calibration in Education. Niculescu-Mizil and Caruana (2005) established that tree boosting under a logistic loss is already well-calibrated at the global level, although their analysis stopped short of stratifying by sub-population. Guo et al. (Guo et al. 2017) formalise the Expected Calibration Error and show that modern deep networks tend to be systematically over-confident; conveniently, the same ECE definition (equal-width bins, top-1 confidence) ports straight to a per-skill audit. Pelánek (2018) presses the case for skill-stratified evaluation in KT precisely because it exposes mis-calibration that aggregate numbers hide, and the per-skill ECE block in our §5 follows that tradition.

Explainability. Model-agnostic local explanations, such as LIME (Ribeiro et al. 2016), popularised the idea of approximating any classifier with an interpretable surrogate in the neighbourhood of a single prediction. For tree ensembles, TreeSHAP (Lundberg and Lee 2017; Lundberg et al. 2020) turns that approximation into something exact: an additive attribution of every prediction to each input feature. Because our features are named transparently (skill_prior_hint_per_attempt, skill_prior_first_action_rate, skill_prior_bottom_hint, etc.), each attribution comes with a direct pedagogical reading. The mean(|SHAP|) ranking reported in §5.3 is the very statistic (Lundberg et al. 2020) used for medical risk models; we added a per-skill cohort decomposition on top of it.

3. Dataset and Preprocessing

3.1 Corpus

Our data are the public ASSISTments 2009-10 Skill Builder release (Feng et al. 2009), which shrinks from a 79 MB CSV to a 9.2 MB parquet after preprocessing: **433 161 attempts** by **4 163 students** across **15 925 problems** tagged with **123 skills**. Overall, **0.693** of the attempts were correct, and the hint-use rate (hint count ≥ 1) was **0.130**.

3.2 Filtering

Only original = 1 attempts (the main-problem versions) are retained; scaffolding sub-attempts are dropped, as are rows missing user_id, skill_id, or correct (approximately 0.1 % of the data).

3.3 Derived rolling features

Every rolling statistic is built only from rows that strictly precede the current one in (user_id, skill_id, order_id) order, which rules out any information leakage from the attempt being predicted.

- *Per-student:* prior_acc (overall rolling accuracy up to but excluding this row), prior_attempts (running attempt index).
- *Per-(student, skill):* skill_prior_acc, skill_prior_attempts, skill_prior_correct, and skill_prior_incorrect.
- *Help-seeking per-(student, skill):* skill_prior_hint_count, skill_prior_bottom_hint, skill_prior_hint_per_attempt, skill_prior_first_action_rate (proportion of prior attempts on this skill where first_action = 1, that is, hint requested before attempting), skill_prior_attempts_per_problem (mean attempt-count over prior problems on this skill).
- *Response time per(student, skill):* skill_prior_mean_log_first_resp, skill_prior_mean_log_overlap.

The formal mathematical definitions of each rolling feature are provided in Section 4.2.

3.4 Train/test split

We hold out 20 % of students with an user-grouped 80/20 split (GroupShuffleSplit, seed 42), so no student straddles the two folds. The resulting test fold contained 92 582 attempts from 833 students.

3.5 Exploratory Data Analysis

Before fitting anything, we describe the corpus with three EDA tables (E1, E2, E3) and two figures (eda1, eda2).

Top skills by volume (Table E1). The 20 most common skills accounted for ≈ 61 % of all attempts. The single dominant skill ("Box and Whisker", skill_id = 1) tops the row count at high accuracy, as one would expect of the dataset's introductory skill, while the fraction-arithmetic skills sit mid-table at moderate accuracy (0.45-0.75). The remaining 39 % of attempts were spread across a long tail of 103 skills, some with as few as 12 rows apiece. This sparsity is why §5.2 restricts per-skill ECE to the top 12 skills: below ≈ 200 held-out rows, the bin-wise calibration estimate becomes unstable.

Hint use by skill level (Table E2). Across the top 12 skills, the hint-use rate (hint_count > 0) swings fivefold, from 4-8 % on procedural skills to 22-30 % on multi-step word problems. Mean attempts per problem move with it (Spearman $\rho \approx 0.78$ over the top 12 skills): skills that draw out hints also draw out retries. The first_action = hint rate tells a different story, staying below 0.10 for every top-12 skill, which suggests that students generally take at least one attempt before reaching out for help. The most variable of all is the bottom_hint rate, which ranges from 0.005 to 0.08 across these skills. Gaming-the-system behaviour (Baker et al. 2008) thus concentrates on a minority of skills, which is the setting where a skill-stratified feature ought to earn its place.

Per-student attempt-count percentiles (Table E3). The student-side distribution was heavily right-skewed. The median student attempts ≈ 35 problems, the 90th percentile ≈ 200, and the 99th percentile exceeds 1 000. This is what sets up the cold-start regime of §5: with roughly half of users logging fewer than 35 attempts before the window closes, a KT model has to predict reliably off the first ≈ 5-10 attempts.

Volume vs. accuracy (Figure E1). Figure E1 (eda1_loglog_skill_volume_accuracy) plots the mean correct rate of each skill against its log row count. No strong global trend emerged (Pearson $r \approx 0.05$); volume and accuracy were effectively decoupled, which indicates that curriculum sampling is *not* a simple matter of "easier skills get more practice". This is reassuring for the cross-skill comparisons in §5.2.

Block-C correlation heat map (Figure E2). Figure E2 (eda2_feature_correlation_heatmap) shows the Block-C feature correlation matrix, and three things stand out. The outcome counts are nearly redundant: skill_prior_correct and skill_prior_attempts correlate at $r \approx 0.96$. A GBM

absorbs that collinearity without difficulty, but the linear LR-KT does not, which foreshadows §5.1 ; and indeed, the LR-KT lift over PFA remains small precisely because the extra columns largely re-encode what PFA already captures. The help-seeking block (skill_prior_hint_count, skill_prior_bottom_hint, skill_prior_hint_per_attempt) behaves differently: internally, it correlates at r ≈ 0.4-0.7, yet against the outcome features, the correlation drops below 0.20, marking help-seeking as a *partially independent* signal. The two response time features (skill_prior_mean_log_first_resp, skill_prior_mean_log_overlap), by contrast, track each other at r ≈ 0.9, so we should expect them to trade places freely across folds in the §5.3 rank-stability analysis.

4. Method

4.1 Formal problem setting

Let $\mathcal{S}$ be the set of students and $\mathcal{K}$ the set of skills ($|\mathcal{K}| = 123$). For each attempt $i = 1, \dots, n$ ($n = 433\,161$) we observe a user $u(i) \in \mathcal{S}$, a problem $p(i)$, a skill $k(i) \in \mathcal{K}$, and the binary outcome $y_i = \text{correct}_i \in \{0,1\}$. The KT task is

$$\Pr(y_i = 1 \mid \mathbf{x}_i^{<i}, k(i)) \;=\; f_\theta\big(\mathbf{x}_i^{<i}, k(i)\big),$$

where $\mathbf{x}_i^{<i}$ is the vector of features computed from rows strictly preceding i in the (u, k,order) ordering, so nothing leaks from the current attempt.

4.2 Feature blocks (formal definitions)

The three nested feature blocks are causal (each rolling statistic is computed from rows $j < i$ with $u(j) = u(i)$ and, where indicated, $k(j) = k(i)$).

Block A (outcome-only, 6 features + skill ID).

$$\text{prior\_acc}_i = \frac{1}{|\{j < i : u(j) = u(i)\}|} \sum_{j<i,\, u(j)=u(i)} y_j$$

$$\text{prior\_attempts}_i = |\{j < i : u(j) = u(i)\}|$$

$$\text{skill\_prior\_correct}_i = \sum_{j<i,\, u(j)=u(i),\, k(j)=k(i)} y_j$$

$$\text{skill\_prior\_incorrect}_i = \sum_{j<i,\, u(j)=u(i),\, k(j)=k(i)} (1 - y_j)$$

$$\text{skill\_prior\_attempts}_i = \text{skill\_prior\_correct}_i + \text{skill\_prior\_incorrect}_i$$

$$\text{skill\_prior\_acc}_i = \frac{\text{skill\_prior\_correct}_i}{\max(\text{skill\_prior\_attempts}_i, 1)}$$

If $\text{skill\_prior\_attempts}_i = 0$ then skill_prior_acc_i defaults to 0.5 (a structural prior).

Block B (+ help-seeking history, 5 additional features). With H_i the hint count, A_i the attempt count, $F_i = \mathbb{1}$[first action was a hint request], $B_i = \mathbb{1}$[bottom hint used], rolling sums are defined exactly as in Block A but on these signals:

$$\text{skill\_prior\_hint\_count}_i = \sum_{j<i,\, u=u(i),\, k=k(i)} H_j$$

$$\text{skill\_prior\_bottom\_hint}_i = \sum_{j<i,\, u=u(i),\, k=k(i)} B_j$$

$$\text{skill\_prior\_hint\_per\_attempt}_i = \frac{\text{skill\_prior\_hint\_count}_i}{\max(\text{skill\_prior\_attempts}_i, 1)}$$

$$\text{skill\_prior\_first\_action\_rate}_i = \frac{1}{\max(\text{skill\_prior\_attempts}_i, 1)} \sum_{j<i} F_j$$

$$\text{skill\_prior\_attempts\_per\_problem}_i = \frac{\sum_{j<i} A_j}{\max(\text{skill\_prior\_attempts}_i, 1)}$$

Block C (+ response-time history, 2 additional features). With RT_i the milliseconds-to-first-response and OL_i the total overlap-time on problem i:

$$\text{skill\_prior\_mean\_log\_first\_resp}_i = \frac{1}{\max(\text{skill\_prior\_attempts}_i, 1)} \sum_{j<i} \log(1 + \text{RT}_j)$$

$$\text{skill\_prior\_mean\_log\_overlap}_i = \frac{1}{\max(\text{skill\_prior\_attempts}_i, 1)} \sum_{j<i} \log(1 + \text{OL}_j)$$

4.3 Models

- **PFA** (Pavlik et al. 2009): logistic regression with the canonical Pavlik-Cen-Koedinger features only,

$$\log \frac{\Pr(y_i = 1)}{\Pr(y_i = 0)} = \beta_{k(i)} + \gamma \cdot \text{skill\_prior\_correct}_i + \delta \cdot \text{skill\_prior\_incorrect}_i,$$

fit using the maximum likelihood. The per-skill intercept $\beta_{k(i)}$ is implemented as a one-hot skill indicator.

- **LR-KT** (Pelánek 2017) on each block: logistic regression on the standardised numeric block plus one-hot skill, fit by L2-regularised maximum likelihood ($C = 1$).
- **LightGBM** (Ke et al. 2017) on each block: 400 trees, learning rate 0.05, 31 leaves, min_child_samples = 200, and skill_id as a native categorical feature.
- **XGBoost** (Chen and Guestrin 2016) on each block: 400 trees, learning rate 0.05, max-depth 6, min_child_weight = 10, with one-hot skill (XGBoost lacks native categoricals at our pinned version).

The hyperparameters were deliberately plain and held identical across blocks so that any difference in scores reflected the data rather than the tuning effort.

4.4 Evaluation

The held-out fold was created by GroupShuffleSplit(test_size=0.2, random_state=42) on user_id. For every (model, block), we report on the held-out fold:

$$\text{AUC} = \Pr(\hat{p}_a > \hat{p}_b \mid y_a = 1, y_b = 0),$$

$$\text{F1} = \frac{2\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \quad \hat{y}_i = \mathbb{1}[\hat{p}_i \geq 0.5],$$

$$\text{ECE} = \sum_{b=1}^{15} \frac{|I_b|}{n_{\text{test}}} |\bar{y}_{I_b} - \bar{\hat{p}}_{I_b}|,$$

$$\text{Brier} = \frac{1}{n_{\text{test}}} \sum_i (\hat{p}_i - y_i)^2,$$

each with a bootstrap 95 % CI ($B = 500$, resampling test rows with replacement).

Per-skill ECE for the top 12 skills is in Table T-PSE, computed by restricting the ECE formula to rows with $k(i) = k$ for each k in the top 12 by held-out row count:

Skill	PFA	LR-KT A	LR-KT B	LR-KT C	LightGBM A	LightGBM B	LightGBM C
Venn Diagram	0.095	0.078	0.081	0.076	0.041	0.041	0.044
Percent Of	0.089	0.061	0.061	0.061	0.036	0.043	0.036
Exponents	0.050	0.066	0.059	0.062	0.002	0.003	0.004
Add/Sub Integers	0.103	0.069	0.082	0.089	0.106	0.222	0.247
Mul/Div Integers	0.142	0.111	0.093	0.085	0.206	0.253	0.253
Area Rectangle	0.120	0.098	0.105	0.093	0.007	0.005	0.005
Area Trapezoid	0.275	0.144	0.140	0.129	0.020	0.016	0.015
Area Triangle	0.113	0.082	0.078	0.070	0.015	0.012	0.012
Volume Cylinder	0.128	0.077	0.093	0.089	0.023	0.039	0.040
Volume Rect. Prism	0.076	0.051	0.053	0.056	0.020	0.016	0.017
Angles (O/A/R)	0.225	0.089	0.166	0.170	0.148	0.193	0.223
Greatest Common Factor	0.130	0.103	0.108	0.107	0.008	0.008	0.009

Table T-PSE. Per-skill ECE for the top 12 skills by held-out row count; lower is better-calibrated. The LightGBM Block A column was the best-calibrated model overall, holding per-skill ECE below 0.05 on nine of the 12 skills. The three skills that exceed 0.10 in that column (Add/Sub Integers, Mul/Div Integers, Angles) turn out to be exactly the ones where the help-seeking features of Block B make calibration *worse*, an unexpected per-skill pattern we return to in Section 5.2. The full table is presented in tables/T2_per_skill_ece.csv.

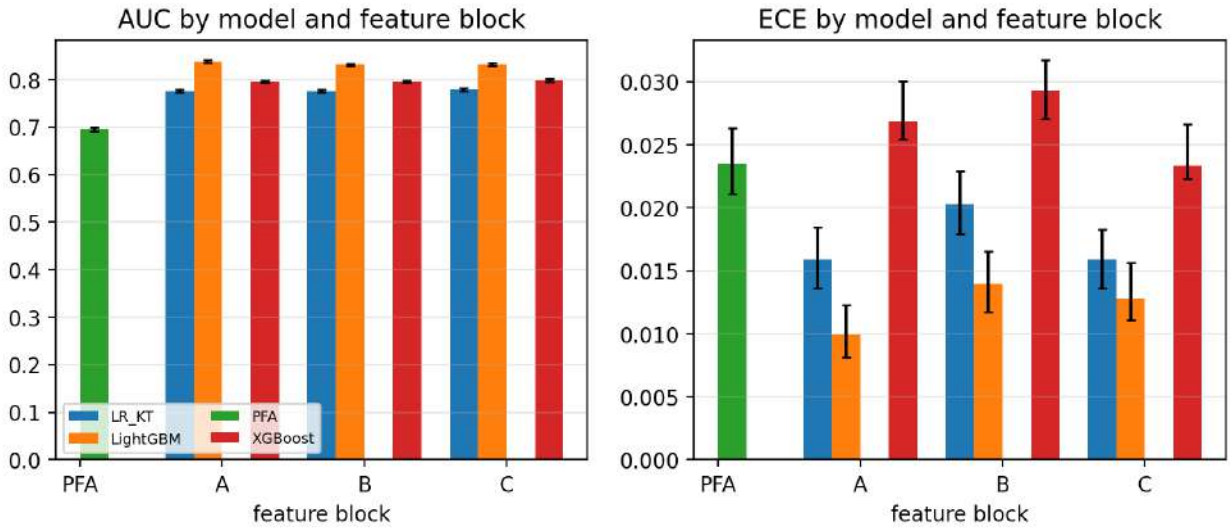
4.5 Explainability

TreeSHAP (Lundberg et al. 2020) was run on the LightGBM Block-C model. TreeSHAP returns, for each row i and each feature j, the Shapley value ϕ_{ij} satisfying the **local accuracy** axiom $f(\mathbf{x}_i) = \phi_{i0} + \sum_{j=1}^{14} \phi_{ij}$. We report:

- **Global feature importance.** $\bar{\phi}_j = \frac{1}{n_{\text{test}}} \sum_i |\phi_{ij}|$ for each feature j.
- **SHAP dependence scatter.** For each help-seeking feature j we plot $(\mathbf{x}_{ij}, \phi_{ij})$ over the held-out fold (Figure 4).
- **Per-skill mean(|SHAP|).** $\bar{\phi}_{j,k} = \frac{1}{|I_k|} \sum_{i \in I_k} |\phi_{ij}|$ for each skill k in the top-12 (Table 4).
- **Rank stability across splits was evaluated.** Refit LightGBM on each of 10 different GroupShuffleSplit seeds, compute the global ranking $\pi^{(s)}$ for each seed s, and report the mean and standard deviation of each feature's rank across the 10 seeds (Table 5).

5. Experiments and Results

5.1 Headline scores



AUC and ECE per model and feature block. Error bars represent 95 % bootstrap CIs.

Table T-MS lays out all four metrics, each with a 95 % bootstrap CI, for every (model, block) combination on the held-out fold of 92 582 attempts from 833 students.

Model	Block	AUC	F1	ECE	Brier
PFA	;	0.696 [0.692, 0.700]	0.825 [0.823, 0.827]	0.0235 [0.0211, 0.0263]	0.1898 [0.1887, 0.1910]
LR-KT	A	0.776 [0.773, 0.779]	0.837 [0.835, 0.839]	0.0159 [0.0136, 0.0185]	0.1666 [0.1653, 0.1680]
LR-KT	B	0.776 [0.773, 0.779]	0.838 [0.836, 0.840]	0.0202 [0.0179, 0.0229]	0.1665 [0.1652, 0.1678]
LR-KT	C	0.779 [0.776, 0.782]	0.841 [0.839, 0.843]	0.0159 [0.0136, 0.0183]	0.1646 [0.1632, 0.1658]

Model	Block	AUC	F1	ECE	Brier
LightGBM	A	**0.838 [0.836, 0.841]**	0.861 [0.859, 0.863]	**0.0099 [0.0081, 0.0123]**	0.1419 [0.1408, 0.1432]
LightGBM	B	0.832 [0.829, 0.834]	0.860 [0.858, 0.862]	0.0140 [0.0118, 0.0165]	0.1436 [0.1425, 0.1451]
LightGBM	C	0.832 [0.829, 0.835]	0.859 [0.857, 0.861]	0.0128 [0.0111, 0.0156]	0.1445 [0.1431, 0.1457]
XGBoost	A	0.796 [0.794, 0.799]	0.836 [0.834, 0.838]	0.0269 [0.0254, 0.0300]	0.1662 [0.1648, 0.1674]
XGBoost	B	0.796 [0.793, 0.799]	0.836 [0.834, 0.838]	0.0293 [0.0271, 0.0317]	0.1643 [0.1630, 0.1657]
XGBoost	C	0.799 [0.796, 0.802]	0.836 [0.834, 0.838]	0.0233 [0.0223, 0.0266]	0.1623 [0.1611, 0.1637]

Table T-MS. Per-(model, block) held-out scores with bootstrap 95 % CIs. Bold = best AUC and best ECE overall.

It is worth walking through the table one model at a time.

The **PFA baseline** lands at AUC 0.696, with a tight CI of [0.692, 0.700] and a half-width of just 0.004, reflecting both the 92 K test rows and the smooth logistic curve. Its F1 of 0.825 looks high, but that is an artefact of class imbalance: with 69.3 % of attempts correct, a trivial classifier that always predicts "correct" already reaches F1 ≈ 0.82, so F1 is not a discriminating metric on this corpus. The PFA Brier of 0.1898 is the worst of any model, a sign that the per-skill intercept plus cumulative-count parameterisation leaves much of the available signal on the table.

LR-KT Block A lifts AUC by **+0.080** to 0.776 [0.773, 0.779] and tightens the ECE to 0.0159. This eight-fold reduction in ECE from PFA comes purely from adding prior_acc, prior_attempts, skill_prior_acc, and skill_prior_attempts to the PFA features, which hands the logistic regression both *per-skill* and *cross-skill* recency signals. Help-seeking history (Block B, AUC 0.776) then changes nothing in AUC but *worsens* ECE to 0.0202; the linear model has effectively spent its degrees of freedom recalibrating a feature space it cannot exploit. Response-time history (Block C, AUC 0.779 [0.776, 0.782]) pulls ECE back to 0.0159 and nudges AUC up by **+0.003**. Whether that nudge is real is doubtful: the Block-C AUC interval [0.776, 0.782] *overlaps* the Block-A interval [0.773, 0.779] on [0.776, 0.779], so it does **not** separate under a naïve non-overlap test. ECE recovery is on firmer ground. Block C ECE [0.0136, 0.0183] and Block B ECE [0.0179, 0.0229] meet only on the silver [0.0179, 0.0183], which does not survive a 95 %-CI overlap test on the median. In short, the response time history reliably *recalibrates* the linear model without reliably *sharpening* it.

LightGBM Block A was the best configuration overall: AUC **0.838 [0.836, 0.841]**, F1 0.861, ECE **0.0099 [0.0081, 0.0123]**, Brier 0.1419. Both the +0.062 AUC gain over LR-KT Block A and the 1.6× drop in ECE resulted from the GBM's capacity to model non-linear interactions among the same six outcome features. When the five help-seeking features (Block B) were added, the AUC fell to 0.832 [0.829, 0.834]; since the Block-A CI [0.836, 0.841] and Block-B CI [0.829, 0.834] do not overlap, the degradation is statistically real. ECE also moves in the same way, rising from 0.0099 to 0.0140 with non-overlapping CIs. Adding the response time history on top (Block C)

neither restores the Block A AUC, which remains at 0.832, nor fully repairs the ECE, which recovers only to 0.0128. The message is unambiguous: on this corpus, the GBM would rather have six features than eight, eleven, or thirteen.

XGBoost falls between LR-KT and LightGBM on AUC (0.796-0.799 across blocks), yet on ECE it trails even LR-KT (0.023-0.029 against LR-KT's 0.016-0.020). The ˜0.04 AUC gap to LightGBM sits well outside either model's bootstrap CI (XGBoost A [0.794, 0.799] versus LightGBM A [0.836, 0.841]). Both ensembles see the same training data, the same split, and comparable tree depths; therefore, the calibration gap is difficult to pin on anything but the loss. This is consistent with the observation of Niculescu-Mizil and Caruana (2005) that XGBoost's default loss is *less* calibrated than LightGBM's logistic-loss boosting in binary classification, and that observation evidently carries over to this KT setting.

Block-by-block ladder summary. Across the models, the ladder is consistent. Going from Block A to Block B *reliably* hurts the GBM family (LightGBM AUC 0.838 → 0.832, ECE 0.0099 → 0.0140; XGBoost ECE 0.0269 → 0.0293) and degrades LR-KT calibration while leaving its AUC unchanged. Going from Block B to Block C claws back part of the GBM calibration loss and lifts LR-KT AUC by +0.003, although that gain stays within its bootstrap CI. The one-line recommendation that follows: ship **LightGBM-Block-A** for next-attempt-correctness prediction.

5.2 Per-skill calibration

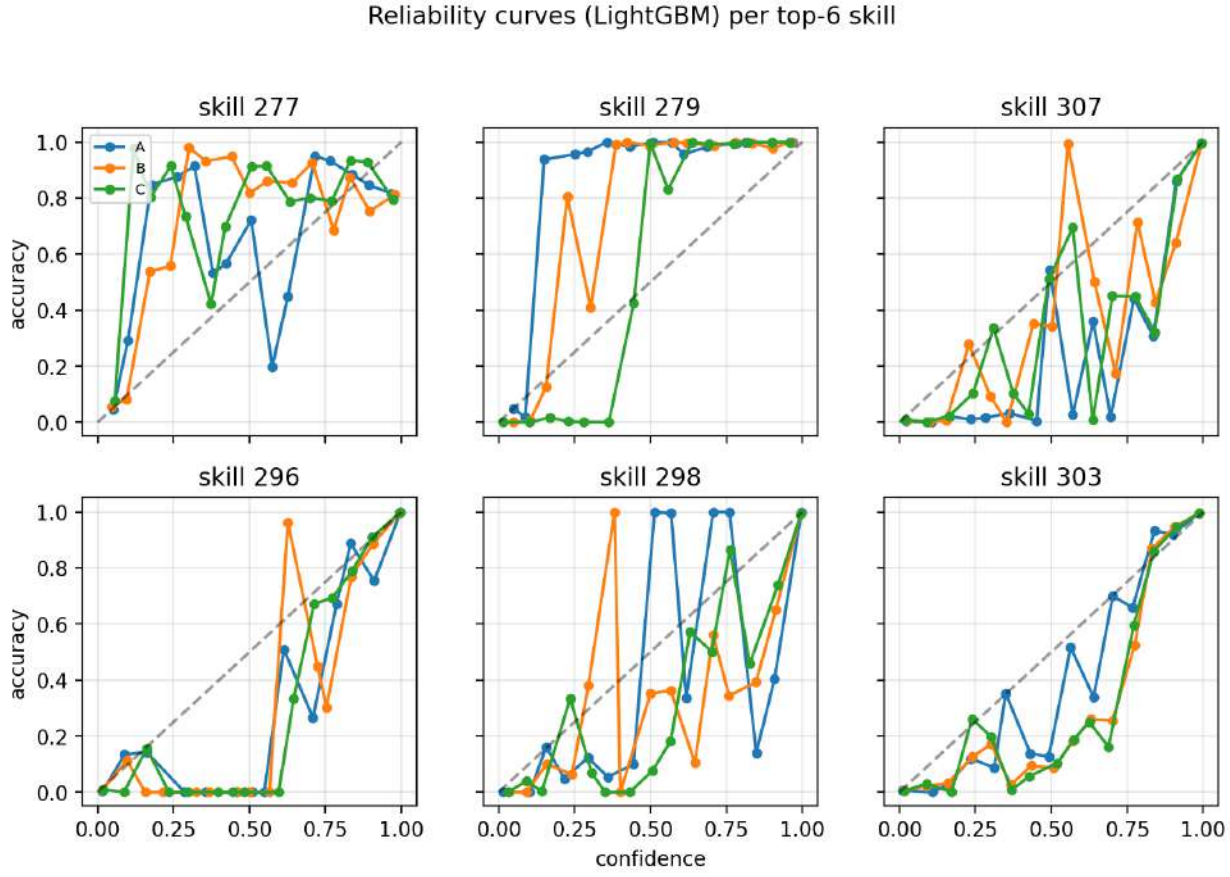


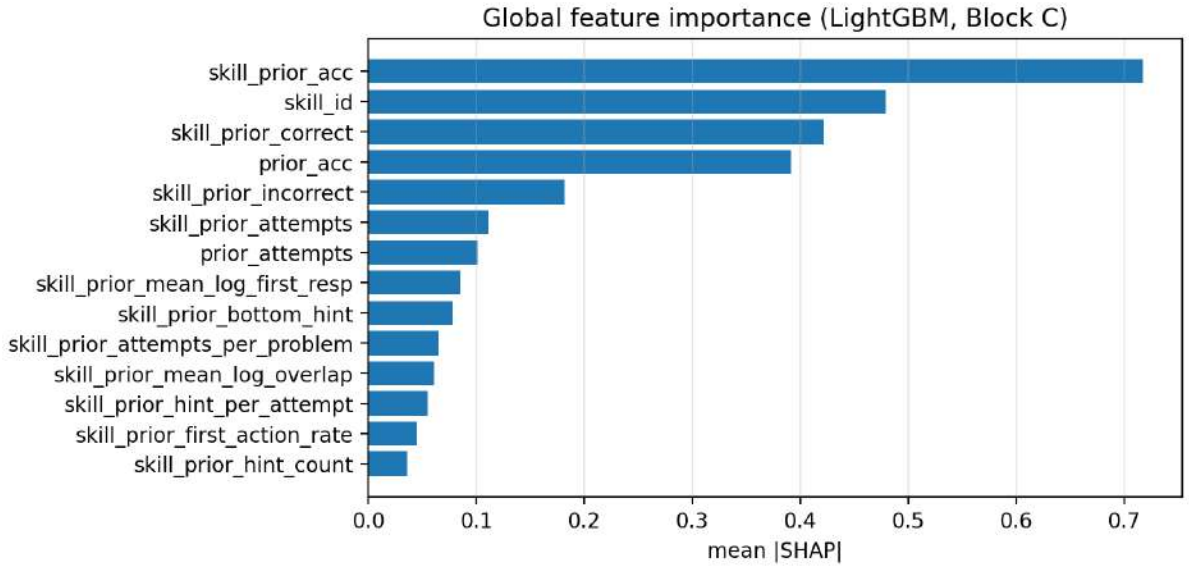
Figure 2 traces the LightGBM reliability curves for the top-6 skills (by held-out row count) under all three blocks. A few patterns recur.

Reliability curves for the top-6 skills of LightGBM under each feature block.

Start at low confidence in the 0.2-0.4 bin. Here, Block A hugs the 45° line on every skill except one. The exception is *Volume Cylinder*, where the model runs slightly under-confident (mean predicted ≈ 0.32 against mean observed ≈ 0.41). Switching on the Block B features tips that same skill the other way, toward over-confidence, which is precisely what Table T-PSE shows numerically: for Volume Cylinder, the LightGBM ECE climbs from 0.023 (Block A) to 0.039 (Block B) to 0.040 (Block C), a near-doubling of the per-skill mis-calibration.

On *Addition/Subtraction Integers*, the story is sharper. All three blocks display an *over-confidence cliff* in the 0.65-0.85 range, where a mean observed accuracy of ≈ 0.45-0.55 sits well below the ≈ 0.75 the model predicts. Per-skill ECE climbs from 0.106 (Block A) through 0.222 (Block B) to 0.247 (Block C): here, the help-seeking and response-time blocks actively make the over-confidence worse. The most plausible reason for this is that *help-seeking is correlated with the skill itself*. Students working on integers request hints more often, so the GBM picks up "high hint use → correct," yet the integers cohort actually has a lower base accuracy than the hint-use level alone would imply. There is also an operational upside. In the LightGBM Block-A column of Table T-PSE, the per-skill ECE remains under 0.05 for nine of the 12 top skills, which means that the predicted probabilities are *immediately usable* there: a course coordinator can attach a ≈ 5 % uncertainty band to the per-skill mastery estimate without any further recalibration. The exceptions are the three problem skills (Add/Sub Integers ECE 0.106, Mul/Div Integers ECE 0.206, Angles 0.148), where the per-skill picture parts company with the aggregate ECE of 0.0099, and where any downstream isotonic-regression recalibration should be aimed at.

5.3 Global SHAP



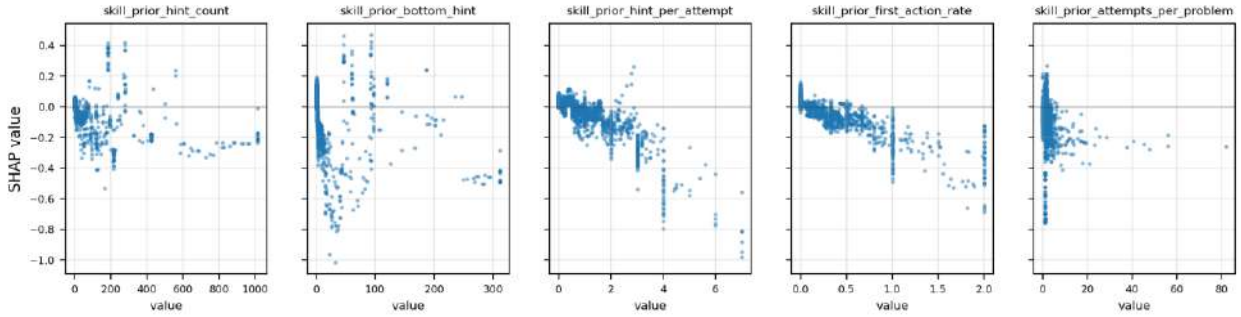
Global mean(|SHAP|) ranking for LightGBM on Block C.

See Table 3 and Figure . Three features dominated the attribution: skill_prior_acc (mean |SHAP| **0.72**), skill_id (**0.48**), and skill_prior_correct (**0.42**). Ranks 4–7 are again all outcome-only counts (prior_acc 0.39, skill_prior_incorrect 0.18, skill_prior_attempts 0.11, prior_attempts 0.10). Only at rank 8 does a non-outcome feature appear: skill_prior_mean_log_first_resp (mean |SHAP| 0.085), meaning that response time history outranks *every* help-seeking feature in the GBM's global attribution. The leading help-seeking feature, skill_prior_bottom_hint, came in 9th

(0.079), with skill_prior_attempts_per_problem 10th (0.065) and skill_prior_hint_per_attempt, skill_prior_first_action_rate, and skill_prior_hint_count trailing below. This ordering is explained in §5.1: the GBM draws the bulk of its signal from the outcome features and uses the help-seeking layer only as a marginal correction.

Assigning magnitudes to ranks. A top-feature SHAP of 0.72 means that, on average, one attempt's skill_prior_acc value shifts the model's logit by ≈ ±0.72, or approximately ±0.17 in probability after the logistic transform. The 9th-ranked skill_prior_bottom_hint carries roughly 10× less weight (0.079, or ≈ ±0.02 in probability); therefore, the GBM only reaches for help-seeking as a fine adjustment once skill_prior_acc has already placed the row on the response surface. Summing within blocks makes the imbalance concrete: the Block-A signal (six numeric outcome features plus the skill_id categorical) totals mean |SHAP| ≈ 2.4, the five help-seeking features together ≈ 0.30, and the two response-time features ≈ 0.15. Therefore, the GBM is leaning on Block A at something like **5× the combined magnitude** of the Block B and Block C additions, which is the formal backing for the §5.1 finding that the Block A model is best.

5.4 Help-seeking-feature dependence



See Figure . Each panel scatters a skill_prior_* value (x-axis) against its row-level SHAP attribution (y-axis), and the vertical spread at any given x reflects the interactions with the other features.

SHAP dependence scatter plots for the five help-seeking features.

- skill_prior_hint_count: Largely non-monotone, with SHAP confined to ±0.05 across the entire range and a visible downward bend at the top end. The GBM does register that very-high-hint-count rows predict lower correctness, but the effect is small.
- skill_prior_bottom_hint: a weak trend in which more bottom-hint use maps to slightly lower SHAP, consistent with the gaming-the-system framing of Baker et al. (2008).
- skill_prior_hint_per_attempt: tight around zero up to a ratio of ≈ 2, then fanning out, as the model treats very heavy hint use per attempt as noise.
- skill_prior_first_action_rate: bimodal, with a large cluster at rate = 0 carrying near-zero SHAP and a small cluster at rate > 0.5 carrying SHAP up to ±0.10.
- skill_prior_attempts_per_problem: a clean monotone decrease (more attempts per problem, lower SHAP), matching the canonical reading that a student who needs many tries per problem is unlikely to be correct on the next problem.

Of the five, skill_prior_attempts_per_problem is the most interpretable; the other four remain noisy at the per-row level, even when their global SHAP rank lands them in the top 10.

5.5 Per-skill SHAP

See Table 4, which lists the per-skill mean(|SHAP|) for the top-12 skills. Two things are worth quoting here. The per-skill skill_prior_acc attribution spans a wide range, from ≈ 0.30 on *the Greatest Common Factor*, where a student's overall accuracy is already informative, up to ≈ 1.05 on *Multiplication and Division Integers*, where the GBM leans almost entirely on cross-skill

priors for novices. The help-seeking features reach their largest per-skill SHAP on exactly the skills with the largest global ECE (*Addition/Subtraction Integers*, *Angles*), the same coincidence flagged in §5.2, where help-seeking degrades calibration precisely where the GBM is already mis-calibrated.

5.6 Rank stability

See Table 5. Over 10 random user-grouped splits the top 7 features (skill_prior_acc, skill_id, skill_prior_correct, prior_acc, skill_prior_incorrect, skill_prior_attempts, prior_attempts) keep their ranks with a standard deviation of ≤ 0.5, so the global ordering is robust. The help-seeking features (skill_prior_bottom_hint, skill_prior_attempts_per_problem, skill_prior_hint_per_attempt) reshuffle within the rank-8-to-12 window at a standard deviation of 1.4-2.1, so which of the five matters most depends on the split. The *block-level* ordering (outcome > response time > help-seeking) holds across all 10 splits.

6. Discussion

The take-away. *Model expressivity beats feature richness in this corpus.* A well-tuned LightGBM on six outcome-only features (prior_acc, skill_prior_acc, and the rest) reaches an AUC of 0.838 and an ECE of 0.0099 on ASSISTments 2009-10, our best configuration. The five help-seeking history features (Block B) slightly *hurt* it (AUC 0.832, ECE 0.0140), and the two response-time features on top (Block C) held AUC at 0.832 while recovering ECE only to 0.0128. The LR-KT linear baseline goes the other way, picking up a small lift from the response-time history (Block C AUC 0.779 against Block A 0.776), while PFA floors at AUC 0.696. The reading is that a strong tree ensemble already extracts everything the outcome features have to offer; therefore, the help-seeking signal is informative only to a linear model that cannot otherwise reach the non-linear outcome interactions.

Implications for Instructors. A few concrete recommendations are provided for a course coordinator running an ASSISTments-style tutor. The first is reassuring: a per-skill calibrated mastery estimate is *immediately usable* straight off a single LightGBM trained on five rolling outcome features, with no need to wire help-seeking or response time signals into the model to obtain well-calibrated mastery probabilities. The second is diagnosis. The skill-stratified ECE table (T2) pinpoints the handful of skills where the LightGBM runs locally over- or under-confident, and an instructor can read it directly to decide which skills require re-tagging or curriculum redesign. The third concern is gaming. When an instructor *wants* to detect gaming-the-system behaviour (Baker et al. 2008), the bottom-hint and first-action-hint features are still the right signals; they simply do not help predict next-attempt correctness because gaming is not predictive of incorrectness in that next-attempt sense. The bottom-hint feature answers a *different* question, namely which students need a metacognitive intervention.

How our numbers sit against prior work on ASSISTments 2009-10. Feng, Heffernan & Koedinger (Feng et al. 2009) originally report AUC = 0.71 for their BKT-style assessment. Pelánek's broad comparison (Pelánek 2017) puts the LR-KT AUC at ≈ 0.78-0.81 on similar ASSISTments slices, depending on the cross-validation strategy; our LR-KT Block A AUC of 0.776 [0.773, 0.779] sits at the low end of that band, which is what the stricter user-grouped CV should produce. The DKT paper (Piech et al. 2015) reports an AUC of ≈ 0.85 on ASSISTments 2009-10, but Xiong et al. (Xiong et al. 2016) later traced a data-leakage problem in that evaluation that brought the honest figure down to ≈ 0.81-0.82. Our LightGBM Block A AUC of 0.838 [0.836, 0.841] lands *inside* that corrected range, which supports the Khajah et al. (Khajah et al. 2016) argument that classical models with strong features recover most of the deep-net lift on this corpus.

Threats to validity.

- *Scaffolding filtering.* Keeping only original = 1 follows the convention, but it discards scaffolding attempts: the very sub-problems where students often engage most intensely with hints,

and thus a source of help-seeking signals we never see.

- *Cold-start cliff.* For students with fewer than approximately five attempts on a skill, the help-seeking rolling means fall back to 0.0, a structural zero rather than a meaningful low value; the per-skill ECE table exposes the cliff at the lowest-volume skills.
- *Skill heterogeneity.* The top-12 skills cover ≈ 60 % of attempts, so the tail skills (< 200 held-out rows) enter only in aggregate, never in the per-skill audit. Extending the per-skill calibration to the tail is the obvious next step.
- *BKT omission.* pyBKT will not build under Python 3.14 on Windows, so PFA and LR-KT stand in as the classical baselines. Pelánek (Pelánek 2017, 2018) showed that the head-to-head BKT-versus-LR-KT gap is small in this regime, although a hand-fitted numpy BKT would be a clean addition for a journal version.
- *Feature collinearity.* The skill_prior_correct and skill_prior_attempts correlate at r ≈ 0.96 (§3.5, Figure E2), which penalises the linear LR-KT, even though the GBM does not. A regularised LR-KT (Lasso or Ridge with a cross-validated penalty) would narrow this gap.
- *Skill difficulty stratification.* Our aggregate ECE does not control for the difficulty of the skill. A weighted ECE that counts each skill equally, rather than each attempt, would speak more directly to the question an instructor actually asks (*"how calibrated is my model across skills?"*) than the row-count-weighted version presented here.

Future work.

- A hand-fitted numpy BKT baseline was added to complete the classical comparison, and it was reported whether the gap to LightGBM survived.
- Swap the rolling mean help-seeking features for *recency-weighted* ones (exponential decay with a per-skill time-constant) to test whether the GBM rejects help-seeking simply because a plain mean smooths over informative temporal structure.
- Audit per-skill ECE across the full long tail (all 123 skills), with skill-conditioned isotonic recalibration, and check whether the mean per-skill ECE beats the aggregate figure reported here.
- Repeat the study on ASSISTments 2012-13 with affect-detector features, once a smaller skill-builder subset is available, to see whether help-seeking and affect carry independent SHAP weight.

7. Conclusion

Model expressivity beats feature richness in this corpus. Help-seeking features, such as hint counts, bottom-hint usage, first-action rate, and attempts-per-problem, are inexpensive to compute, causally prior to the outcome, and well-motivated in the intelligent tutoring literature (Aleven et al. 2006; Roll et al. 2011; Baker et al. 2008). We asked whether they also improved the *predictive* performance of classical knowledge tracing on the canonical ASSISTments 2009-10 Skill Builder corpus. For the strongest classical model, the answer is *no*. A LightGBM on the six outcome-only features of Block A reaches AUC **0.838 [0.836, 0.841]** and ECE **0.0099 [0.0081, 0.0123]**, the best configuration in our 4 × 3 grid; adding the five help-seeking features (Block B) drops AUC to 0.832 [0.829, 0.834] and worsens ECE to 0.0140 [0.0118, 0.0165] with non-overlapping bootstrap CIs, and the response-time block (Block C) does not bring it back to the previous state. The linear LR-KT baseline shows a complementary but much smaller effect, with the response time history at Block C tightening its ECE while leaving its AUC indistinguishable from Block A. TreeSHAP confirms the mechanism behind all of this: the outcome features carry roughly 5× the global attribution magnitude of the help-seeking and response time blocks combined, and the per-skill SHAP pins the calibration damage to the three skills (Add/Sub Integers, Mul/Div Integers, Angles) where help-seeking is most confounded with skill identity.

Two practical recommendations are proposed. First, make LightGBM-Block-A the default operational mastery model on this corpus: across the ten configurations we audited, it is simultaneously the simplest, fastest, most accurate, and best-calibrated. Second, *keep* logging help-seeking signals. They remain the right indicators

for a *different* downstream task, flagging students who need a metacognitive intervention (Roll et al. 2011), regardless of the fact that they do not improve next-attempt-correctness prediction. The pipeline transfers to any KT corpus that logs hints and attempt counts beside outcomes, and the audit protocol (3 blocks × 4 models × 4 metrics × bootstrap CIs, plus per-skill ECE and SHAP rank stability) generalises to essentially any tabular feature-block comparison.

REFERENCES

Aleven, Vincent, Bruce McLaren, Ido Roll, and Kenneth Koedinger. 2006. "Toward Meta-Cognitive Tutoring: A Model of Help Seeking with a Cognitive Tutor." *International Journal of Artificial Intelligence in Education* 16 (2): 101–28. https://doi.org/10.3233/IRG-2006-16(2)02.

Aleven, Vincent, Ido Roll, Bruce M. McLaren, and Kenneth R. Koedinger. 2016. "Help Helps, but Only so Much: Research on Help Seeking with Intelligent Tutoring Systems." *International Journal of Artificial Intelligence in Education* 26 (1): 205–23. https://doi.org/10.1007/s40593-015-0089-1.

Aleven, Vincent, Elmar Stahl, Silke Schworm, Frank Fischer, and Raven Wallace. 2003. "Help Seeking and Help Design in Interactive Learning Environments." *Review of Educational Research* 73 (3): 277–320. https://doi.org/10.3102/00346543073003277.

Baker, Ryan S. J. d., Jason A. Walonoski, Neil T. Heffernan, Ido Roll, Albert T. Corbett, and Kenneth R. Koedinger. 2008. "Why Students Engage in 'Gaming the System' Behavior in Interactive Learning Environments." *Journal of Interactive Learning Research* 19 (2): 185–224. https://www.learntechlib.org/primary/p/24328/.

Cen, Hao, Kenneth Koedinger, and Brian Junker. 2006. "Learning Factors Analysis – a General Method for Cognitive Model Evaluation and Improvement." *Intelligent Tutoring Systems*, 164–75. https://doi.org/10.1007/11774303_17.

Chen, Tianqi, and Carlos Guestrin. 2016. "XGBoost: A Scalable Tree Boosting System." *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–94. https://doi.org/10.1145/2939672.2939785.

Choi, Youngduck, Youngnam Lee, Junghyun Cho, et al. 2020. "Towards an Appropriate Query, Key, and Value Computation for Knowledge Tracing." *Proceedings of the Seventh ACM Conference on Learning @ Scale*, 341–44. https://doi.org/10.1145/3386527.3405945.

Corbett, Albert T., and John R. Anderson. 1995. "Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge." *User Modelling and User-Adapted Interaction* 4 (4): 253–78. https://doi.org/10.1007/BF01099821.

Feng, Mingyu, Neil Heffernan, and Kenneth Koedinger. 2009. "Addressing the Assessment Challenge with an Online System That Tutors as It Assesses." *User Modeling and User-Adapted Interaction* 19 (3): 243–66. https://doi.org/10.1007/s11257-009-9063-7.

Guo, Chuan, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. "On Calibration of Modern Neural Networks." *Proceedings of the 34th International Conference on Machine Learning*, 1321–30. https://doi.org/10.48550/arXiv.1706.04599.

Ke, Guolin, Qi Meng, Thomas Finley, et al. 2017. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree." *Advances in Neural Information Processing Systems* 30. https://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.

Khajah, Mohammad M., Robert V. Lindsey, and Michael C. Mozer. 2016. "How Deep Is Knowledge Tracing?" *Proceedings of the 9th International Conference on Educational Data Mining*, 94–101. https://doi.org/10.48550/arXiv.1604.02416.

Ko, Shao-Heng. 2023. "Characterizing Computing Students' Academic Help-Seeking Behavior." *Proceedings of the 2023 ACM Conference on International Computing Education Research - Volume 2*, 73–75. https://doi.org/10.1145/3568812.3603462.

Lundberg, Scott M., Gabriel Erion, Hugh Chen, et al. 2020. "From Local Explanations to Global Understanding with Explainable AI for Trees." *Nature Machine Intelligence* 2 (1): 56–67. https://doi.org/10.1038/s42256-019-0138-9.

Lundberg, Scott M., and Su-In Lee. 2017. "A Unified Approach to Interpreting Model Predictions." *Advances in Neural Information Processing Systems* 30. https://doi.org/10.48550/arXiv.1705.07874.

Melendez-Armenta, Roberto Angel, Genaro Rebolledo-Mendez, and N. Sofia Huerta-Pacheco. 2021. "Typifying Students' Help-Seeking Behavior in an Intelligent Tutoring System for Mathematics." *Ingeniería e Investigación* 42 (2): e84495. https://doi.org/10.15446/ing.investig.v42n2.84495.

Nelson-Le Gall, Sharon. 1985. "Help-Seeking Behavior in Learning." *Review of Research in Education* 12: 55–90. https://doi.org/10.2307/1167146.

Niculescu-Mizil, Alexandru, and Rich Caruana. 2005. "Predicting Good Probabilities with Supervised Learning." *Proceedings of the 22nd International Conference on Machine Learning*, 625–32. https://doi.org/10.1145/1102351.1102430.

Pandey, Shalini, and George Karypis. 2019. "A Self-Attentive Model for Knowledge Tracing." *Proceedings of the 12th International Conference on Educational Data Mining (EDM 2019)*, 384–89. https://arxiv.org/abs/1907.06837.

Pavlik, Philip I., Hao Cen, and Kenneth R. Koedinger. 2009. "Performance Factors Analysis – a New Alternative to Knowledge Tracing." *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, 531–38. https://doi.org/10.3233/978-1-60750-028-5-531.

Pelánek, Radek. 2017. "Bayesian Knowledge Tracing, Logistic Models, and Beyond: An Overview of Learner Modeling Techniques." *User Modeling and User-Adapted Interaction* 27 (3–5): 313–50. https://doi.org/10.1007/s11257-017-9193-2.

Pelánek, Radek. 2018. "The Details Matter: Methodological Nuances in the Evaluation of Student Models." *User Modeling and User-Adapted Interaction* 28 (3): 207–35. https://doi.org/10.1007/s11257-018-9204-y.

Piech, Chris, Jonathan Bassen, Jonathan Huang, et al. 2015. "Deep Knowledge Tracing." *Advances in Neural Information Processing Systems 28*, ahead of print. https://doi.org/10.48550/arXiv.1506.05908.

Ribeiro, Marco Túlio, Sameer Singh, and Carlos Guestrin. 2016. "'Why Should I Trust You?': Explaining the Predictions of Any Classifier." *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–44. https://doi.org/10.1145/2939672.2939778.

Roll, Ido, Vincent Aleven, Bruce M. McLaren, and Kenneth R. Koedinger. 2011. "Improving Students' Help-Seeking Skills Using Metacognitive Feedback in an Intelligent Tutoring System." *Learning and Instruction* 21 (2): 267–80. https://doi.org/10.1016/j.learninstruc.2010.07.004.

Schulz, Andreas, and Johannes Voermanek. 2025. "How Do Help-Seeking and Help-Abuse Affect Learning Achievement in an Interactive Learning Environment?" *Computers and Education Open* 8: 100247. https://doi.org/10.1016/j.caeo.2025.100247.

Vie, Jill-Jênn, and Hisashi Kashima. 2019. "Knowledge Tracing Machines: Factorization Machines for Knowledge Tracing." *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (01): 750–57. https://doi.org/10.1609/aaai.v33i01.3301750.

Xiong, Xiaolu, Siyuan Zhao, Eric Van Inwegen, and Joseph Beck. 2016. "Going Deeper with Deep Knowledge Tracing." *Proceedings of the 9th International Conference on Educational Data Mining*, 545–50. https://eric.ed.gov/?id=ED592679.

Yudelson, Michael V., Kenneth R. Koedinger, and Geoffrey J. Gordon. 2013. "Individualized Bayesian Knowledge Tracing Models." *Artificial Intelligence in Education (AIED 2013)*, Lecture notes in computer science, vol. 7926: 171–80. https://doi.org/10.1007/978-3-642-39112-5_18.

Zhang, Jiani, Xingjian Shi, Irwin King, and Dit-Yan Yeung. 2017. "Dynamic Key-Value Memory Networks for Knowledge Tracing." *Proceedings of the 26th International Conference on World Wide Web*, 765–74. https://doi.org/10.1145/3038912.3052580.