Beat the Machine: A Gamified Approach to Identifying Systematic Errors in Predictive Models

Akash Kota Raju

Independent Researcher, New York University, New York
ak9381@nyu.edu

Abstract. In predictive modeling, systematic errors—often referred to as unknown unknowns—can lead to critical failures, especially when models are overconfident in their incorrect predictions. This paper presents a gamified framework, Beat the Machine (BTM), which incentivizes the identification of such systematic misclassifications. BTM rewards participants based on error severity, emphasizing high-confidence failures that conventional quality assurance methods tend to overlook. The pro- posed approach is benchmarked against the widely used Stratified Random Sampling method (for a detailed discussion, see Cochran's Sampling Techniques [4]) and demonstrates superior performance in uncovering both the frequency and severity of misclassifications, as supported by im- proved AUC metrics [6]. The results underscore the potential of gamified error detection to enhance model robustness and offer a new paradigm for systematic quality assurance in machine learning applications.

Keywords: machine learning evaluation, model robustness, unknown unknowns, gamified testing, human-in-the-loop, adversarial examples

1 Introduction

This paper addresses a significant yet often overlooked issue in machine learning research and practical implementation. While rarely discussed in academic ML literature, this challenge becomes evident when applying ML solutions in real-world contexts.

We collaborated with a company to develop systems capable of identifying web pages containing problematic content such as "hate speech" (e.g., racist material, antisemitic messages, etc.). The models process web pages as inputs, producing scores that either block such content within the ad ecosystem or generate exposure reports. The aim is to safeguard advertisers whose ads may, despite careful management, appear next to objectionable content. Such associations damage brand reputation and inadvertently provide support to harmful content through ad spending.

A key concern for the company is evaluating any system's efficacy and limitations. Unlike many theoretical ML tasks, practical applications lack a "gold standard" benchmark dataset for hate speech detection. Conventional ML methods rely on a closed-world assumption [10], presuming that all relevant information is captured within the available labeled data. This approach assumes that

P. Ranjan et al. (eds.), *Proceedings of the International Conference on Recent Advances in Artificial Intelligence for Sustainable Development (RAISD 2025)*, Advances in Intelligent Systems Research 196,
https://doi.org/10.2991/978-94-6463-787-8_54

patterns unrepresented or underrepresented in the data simply do not exist. However, such assumptions pose risks when dealing with limited labeled data, rare subgroups [13], and potential biases in data collection. Consequently, models may struggle to predict their performance on new data, leading to challenges with *unknown unknowns*.

In summary, typical ML research assumes data is broadly representative, enabling confident model training and evaluation. Moreover, models are assumed to accurately gauge their own predictive capabilities and provide confidence metrics for different areas. Belief in training data encourages model refinement and confidence-driven decision-making. Essentially, this paradigm focuses on known uncertainties, presuming we understand where models perform poorly.

In practice, however, data collection processes may miss substantial areas of the feature space. Consider hate speech detection: it is a rare and highly varied category—a "disjunctive concept" in ML terms. If specific hate speech examples are absent from training data, the model likely misclassifies them as benign, often with unjustified confidence. Such cases represent "unknown unknowns"—instances that the model not only fails to recognize but does so without awareness of its limitations. This leads to high-risk regions in which models underestimate misclassification costs.

Standard ML evaluation approaches may miss this issue entirely. Cross-validation and AUC scores could suggest nearly flawless performance, potentially misleading ML practitioners into assuming the model's reliability. Such evaluations focus solely on "known unknowns," neglecting the broader impact of hidden vulnerabilities. Ignoring unknown unknowns can lead to unforeseen errors, client dissatisfaction, or public exposure of model weaknesses.

This paper offers a framework emphasizing the significance of unknown unknowns in ML error analysis, distinguishing them from known uncertainties. We introduce a game-based system, Beat the Machine, which leverages crowdsourcing to uncover these hidden issues. Our implementation at industrial scale faced numerous hurdles, which informed subsequent design iterations. Finally, we present experiments demonstrating how focusing on unknown unknowns through this approach reveals previously missed patterns in real-world applications (such as hate speech detection) and highlights systematic errors and data gaps in the original training datasets.

2 Background and Scope

Organizations, both public and private, rely on decisions informed by explicit or implicit models that describe various aspects of reality. Since these models are imperfect, recognizing and understanding their limitations is critical. This knowledge can help (i) enhance the models, (ii) anticipate potential decision-making errors, and (iii) manage associated risks effectively. However, a significant challenge arises when trying to pinpoint where models fall short and how those shortcomings may affect decision-making in complex scenarios. In many

instances, the underlying issue is that we are unaware of the extent of our knowledge gaps—*what we don't know* is beyond our current understanding.

Such lapses can lead to large-scale failures, from terrorist attacks and nuclear accidents to moderate-scale incidents like cybersecurity breaches and routine issues such as inaccuracies in predictive models for credit scoring, fraud detection, and document classification.

This paper specifically examines scenarios where:

- Each decision-making situation can be described by an input description and a corresponding target outcome. We use a predictive model to estimate or score the target for any given scenario. For simplicity, we assume the target is binary and undisputed when known.[1]
- We aim to identify the model's inaccuracies, particularly systematic patterns of errors in areas where the model is confident and estimates a low likelihood of misclassification. For example, are there forms of hate speech that were not considered during model development, resulting in misclassification despite high confidence?
- There exist rare but critical cases or subclasses. Their infrequency often leads to exclusion during system design. In our example, hate speech on the web is relatively rare, and within it, certain types of hate speech may be even less prevalent, such as expressions of hatred toward specific groups like data miners or other marginalized communities.

These characteristics create challenges in identifying model flaws. Merely running the system does not reveal all issues; without knowing the true target value, we cannot gauge the accuracy of model predictions or system decisions.

It is possible to invest in data acquisition to expose model inaccuracies by having humans label random or selectively chosen cases. However, this method has drawbacks. Given the rarity of interest classes (e.g., hate speech), finding positive examples through random sampling is costly. For instance, hate speech comprises less than 0.0001% of ad-supported web pages, with more unusual expressions even less common. To find a single example through random sampling, millions of pages would need labeling. Moreover, multiple labels may be necessary to ensure accuracy [12,9].

In practice, heuristic approaches often guide the identification of potential errors. "Active learning" focuses on finding informative cases [11], typically by choosing cases where models are least certain. Approaches like uncertainty sampling, sampling near decision boundaries, and query-by-committee all prioritize these areas of uncertainty. While useful for model improvement, they address only known uncertainties—instances where model predictions are inherently uncertain.

However, for uncovering *unknown unknowns*, these strategies are ineffective. They focus on known areas of doubt and avoid regions where models are certain but potentially wrong, limiting our ability to identify hidden flaws.

We propose reframing classifier evaluation from a closed-world to an open-world perspective. This paper introduces a system leveraging human input to discover *unknown unknowns*. Our Beat The Machine (BTM) system gamifies this process by rewarding participants for identifying instances where the model fails, with rewards increasing based on the error's magnitude. This system encourages consistent and precise engagement and has been implemented at scale to identify web pages with offensive content. Our findings show that BTM identifies errors that differ significantly from those found through random sampling, revealing large, systematic errors instead of isolated outliers. In essence, BTM allows us to explore *unknown unknowns* and address failures that conventional models cannot yet recognize independently [1].

3 Known-Unknowns

To thoroughly analyze unknown unknowns in predictive modeling, it is necessary to clarify essential concepts. Let x represent an example within a defined problem space X. For classification tasks, every x is associated with a true label, y^-, from a range of possible labels Y. The objective of classification is to build a predictive model, $f(x)$, which generates an estimated label ($y = f(x)$) for each input, aiming for the predicted label y to match as closely as possible with the actual hidden label y^-. Whenever a classification error occurs, it results in a misclassification cost, c_{ij}, where an example with the true label $y^- = y_i$ is incorrectly classified into the category $f(x) = y_j$. Our focus is on models that produce posterior probability distributions over all possible labels, expressed as $f(x) = p(y|x) = \langle p_1, \ldots, p_n \rangle$. These probabilities allow for estimating the expected cost of misclassification.

These probabilities can then guide the selection of a preferred label, such as by choosing the y with the highest probability or, in cost-sensitive situations, by selecting the label with the lowest estimated cost [7]. Notably, emphasizing probability-generating models does not restrict general applicability—techniques exist to convert deterministic models into probability estimators [5].

Equation 3 provides *estimates* of misclassification costs, which may differ from the *actual* costs based on model precision and the reliability of its posterior probability predictions. The discrepancy between estimated and actual costs leads to four potential classification outcomes, as shown in Figure 1.

Instances where the model's estimated cost for misclassification aligns with the true cost lie along the diagonal. In the bottom-left corner, we have situations where the model correctly classifies inputs with high certainty, referred to as "*known knowns*." On the other hand, the top-right corner represents "known unknowns," where the model often makes errors but recognizes its uncertainty.

Definition 1 (Known Unknown).

Let $X' \subset X$ be a subregion and x an instance within X'. Define $\mathrm{ExpCost}(X')$ as the anticipated misclassification cost

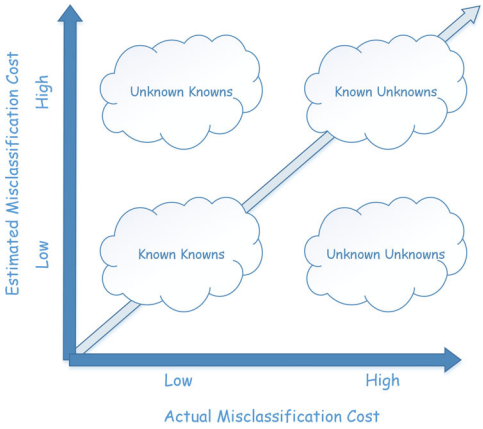

Fig. 1. Model outcomes are categorized as: known knowns (correct/confident), known unknowns (erroneous/uncertain), unknown knowns (correct/uncertain), and unknown unknowns (erroneous/confident).

in X' and ExpCost (X') as the true cost. A known unknown is an instance where both the actual misclassification cost ExpCost (x) and the estimated cost ExpCost (X') are significantly high.

As outlined in Definition 1, known unknowns are commonly encountered in machine learning. The subregion X' highlights an area where errors are more prevalent, a concept leveraged in active learning approaches such as uncertainty sampling [8]. In predictive tasks, "known unknowns" represent cases where errors are anticipated based on the model's confidence level. Addressing these cases might involve opting to "defer" uncertain predictions for expert review [2, 3], necessitating a trade-off between misclassification and deferral costs. However, models in real-world settings may also produce unexpected errors beyond these known uncertainties. For example, consider a hate speech detection system. In practice, it could face complex cases, such as discussions on historical racial issues.[2] These nuanced cases provide a means to evaluate model performance. At the same time, the model might also exhibit overconfidence when
Mis-classifying specific examples. Such occurrences are known as "unknown unknowns."

Definition 2 (Unknown Unknown). According to Definition 1, an " unknown unknown" is an example x' situated outside the uncertainty region, with a low estimated misclassification cost ExpCost (X'), yet a high actual misclassification cost ExpCost (x). In other words, the model is confident in its incorrect prediction.

This definition formalizes "unknown unknowns" as examples far from decision boundaries, where high model confidence leads to incorrect labeling. While isolated outlier cases may fit this definition due to noise or errors, the primary focus is on disjunctive sub-regions of the problem space [13]. These rare but consistent example groups are overlooked in training data and pose real-world risks.

Figure 2 illustrates a common classification scenario involving "unknown unknowns." In the first scenario, we see a linear decision boundary minimizing prediction errors in a training set, with an ϵ-wide uncertainty band around it. This reflects the expected distribution based on random sampling. However, real-world deployment may reveal rare sub-regions previously unseen, posing a risk due to their absence from the training data.

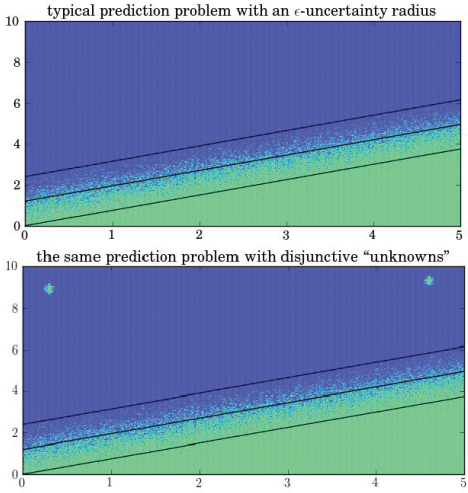

Fig. 2. An example classification scenario: Top—A decision boundary minimizing errors within an inseparable training set, with a ϵ-wide uncertainty region around it. Bottom—The same classifier faced with rare "unknown unknowns," introducing errors beyond the model's uncertainty zone.

Finally, Figure 1 also identifies "*unknown knowns*"—cases where the model shows low confidence and high estimated cost despite mostly correct predictions. Managing these cases often involves perceived inefficiency due to unnecessary interventions. However, addressing unknown knowns is beyond the scope of this paper and remains a subject for future exploration.

4 Outsmarting the Classifier

Evaluating the real-world performance of any automated classification system poses unique challenges. Complexities such as class imbalance and rare disjunctive sub-concepts, as seen with the hate speech classifier described in Section 1, make accurate evaluation particularly challenging and often lead to the emergence of unknown unknowns. Traditionally, a common approach involves sampling output decisions and using human evaluators to verify classification correctness. This allows for estimating error rates across different parts of the classification spectrum. However, given the nature of our problem, this process can be highly inefficient. Firstly, when classifications are generally accurate, human reviewers rarely encounter errors without focused sampling. Secondly, in cases with significant class imbalance, the majority of observed errors are misclassifications of majority class examples into the minority class. This becomes problematic since errors in highly imbalanced scenarios, where the minority class often incurs a higher cost—like the case of hate speech detection—have a greater impact. Lastly, identifying rare disjunctive sub-concepts is difficult; such occurrences might be as rare as 1 in a million. This rarity complicates the task of identifying misclassified minority examples.

Consider a scenario of detecting web pages with hate speech content. If the classifier has a relatively high accuracy, correctly labeling 95% of examples, identifying pages with misclassified hate speech becomes difficult. In a random sample, most pages would be accurately labeled as benign. Even under the generous assumption that 0.1% of pages on the internet contain hate speech, locating a single false negative (i.e., hate speech passing as benign) would require reviewing roughly 20,000 pages, while encountering about 1,000 false positives in the process. While it might be tempting to dismiss such issues as negligible, these errors can accumulate when filtering billions of pages, making seemingly rare errors significant in absolute terms. Additionally, even isolated cases can lead to considerable repercussions, such as a company's reputation suffering if it inadvertently supports offensive content through advertising. Unknown unknowns are especially concerning because client expectations aren't managed effectively, and comprehensive contingency plans often don't exist. Rather than passively waiting for these hidden errors to surface, we can actively work to uncover them. We propose a system that leverages human intelligence through crowdsourcing to identify these "unknown unknown" cases. This approach is akin to employing "white hat" hackers to find security vulnerabilities in an organization's systems. In this instance, human participants are tasked with finding examples that "outsmart" the classifier.

4.1 Developing the Outsmarting Task

To better understand the Outsmarting the Classifier process, we present a series of progressively advanced designs, refining the approach as new challenges emerge.

Initial Approach: The simplest version involved asking users to uncover unknown unknowns, essentially finding examples where the model would fail. Participants were invited to submit URLs they believed would be misclassified by the system. To motivate engagement, each submission earned a small reward, while a misclassified URL earned a larger bonus. (Specifically, we offered a nominal rate of 1 cent for every 5 URLs submitted, with a 50-cent bonus for any URL that was misclassified.)

However, this setup presented a significant issue: how could the system validate whether a URL truly challenged the model's accuracy? The goal was to find instances where the model erred without realizing it! To verify misclassifications, we enlisted trusted human evaluators to label the URLs submitted by participants, then compared these human-assigned labels to the model's predictions. To mitigate potential abuse, participants for the Outsmarting Task were recruited via Amazon Mechanical Turk, while trusted evaluators were sourced and trained through platforms like oDesk in automated settings or were student interns for controlled experiments. Despite the initial simplicity, this approach was flawed for several reasons.

The primary obstacle was the lack of immediate interaction. While users could submit URLs they thought would outsmart the model, they had to wait for human evaluation to know whether their submissions were successful. This delay—ranging from a few minutes to hours—lowered the overall user experience by reducing engagement and denying participants instant feedback on their performance.

Implementing Real-Time Classification Feedback: To address this is- sue, we upgraded the system to immediately analyze submitted URLs and in- form users of the model's prediction. For example, the system might display, "The model classifies this URL as containing hate speech. Do you agree?" This allowed users to decide whether they believed the model's output was incorrect before submitting the URL for further review. Upon submission, participants earned provisional bonus points (convertible to cash), which would be confirmed after human verification.

Although this enhancement improved engagement, it did not fully align with our intended incentives. Users found it easier to spot majority-class examples (e.g., benign pages) mistakenly flagged as containing hate speech. As a result, rather than surfacing high-impact errors, users frequently submitted errors detectable by simply examining the model's positive classifications. Given the im- balance in class distribution, most errors involved benign pages wrongly categorized as containing hate speech. Our key goal, however, was to detect cases of hate speech misclassified as benign, particularly among unknown unknowns. Furthermore, some participants attempted to game the system by submitting random URLs while consistently claiming the classifier was incorrect, even when it was not.

Task Segmentation by Class: To address existing challenges, we broke the task down into two specific subtasks: (1) identifying pages from the minority class that are incorrectly categorized under the majority class (e.g., offensive content

being marked as benign) and (2) identifying benign pages that are misclassified as containing offensive content. This segmentation helped simplify the task and made it more intuitive for participants. Moreover, it enabled rapid dismissal of irrelevant submissions. For instance, if users were instructed to find misclassified hate speech content, pages that the classifier strongly marked as hate speech could be quickly rejected. In the original design, participants were motivated to falsely mark these pages as "non-hate speech" with the hope that human reviewers would accept their assessments.

Enhancing Incentive Structures: In our final implementation, we refined the incentive mechanism to distinguish between users who identified major errors ("unknown unknowns") and those who located more predictable errors ("known unknowns"). Instead of offering a flat bonus for misclassified URLs, rewards were adjusted based on the estimated cost of misclassification, inferred through model confidence scores. High-confidence errors, where the model's certainty was near 100%, resulted in the highest user rewards, as they reflect more serious misclassification mistakes. Conversely, lower-confidence errors with higher predicted costs received proportionally smaller bonuses. Users were also incentivized for providing examples where the model was accurate but uncertain. For example, if the model was 60% confident that a page contained hate speech, and it was indeed true, a minor reward was given.

5 Experimental Evaluations

Section 3 outlined the concept of unknown unknowns, while Section 4 described how our gamified system motivates users to identify these instances in predictive models. To assess the effectiveness of BTM, we examined three primary questions:

- How efficiently does BTM uncover errors?
- Does BTM primarily identify isolated cases of unknown unknowns or broader patterns within the model's predictions?
- Can the identified errors improve model accuracy?

We conducted tests using BTM on two classification models: one for detecting web pages with hate speech and another for identifying adult content. The

experiments utilized the configurations outlined earlier (0.2 cents base pay per task, with a maximum of 50 cents for error-triggering URLs).

Comparing with Stratified Random Sampling:

In this domain, the standard approach for quality assurance involves using stratified random sampling of model outputs. Random sampling across predictions is often uninformative due to the classifier's high accuracy and imbalanced distributions, resulting in most predictions being accurate and non-problematic. Consequently, human evaluators focus on random samples divided by model confidence intervals. Specifically, the confidence range (0,1) is divided into k equal-width intervals, and a sample of N URLs is drawn, with $\frac{N}{k}$ from each interval. This approach over-samples lower-confidence predictions, increasing the likelihood of finding errors and providing insights into model calibration based on the distribution of positive and negative examples in each bin. However, it mainly reveals "known unknowns." Our comparison involved contrasting this method with BTM's approach. It is crucial to note that these methods are complementary; each designed to assess distinct aspects of model performance.

For the adult content classifier, human evaluators identified errors in 16% of inspected cases using stratified sampling—a significant increase over the model's natural error rate. By contrast, BTM yielded errors in over 25% of examined cases. For the hate speech classifier, stratified sampling detected errors in 9% of cases, while BTM uncovered errors in 27% of examined URLs. These results suggest that BTM may be more effective at surfacing problematic cases than standard quality checks. However, simply increasing the sampling of low-confidence predictions would reduce unknown unknowns discovery, as it would focus predominantly on known issues. In extreme cases, sampling solely from low-confidence regions would maximize error discovery but at the cost of limiting the identification of new, unanticipated errors.

Error Severity Analysis: Figures 4 and 5 illustrate the distribution of errors identified in models used for hate speech and adult content detection. Errors detected by the BTM approach are represented by blue bars, while those found through stratified evaluation appear in red. The horizontal axis is divided into four levels indicating error severity, from minimal mistakes on the left (where zero indicates no error) to the most significant misclassification on the right, marked at 1000, which represents complete confidence by the model in a wrong prediction. The extreme right end denotes "unknown unknowns."

Both categories reveal a consistent pattern: BTM uncovers a higher proportion of serious errors—"unknown unknowns." Among the errors identified by BTM, 25% were severe cases, where the model exhibited high confidence in an incorrect decision (for example, wrongly categorizing harmful content as benign).

To summarize, BTM not only detects a greater number of problematic predictions than stratified testing but also identifies a considerable amount of previously unrecognized errors. These would otherwise go unnoticed unless highlighted by a major incident. On the other hand, most errors detected through

stratified testing tend to cluster around the model's decision boundary, as expected.

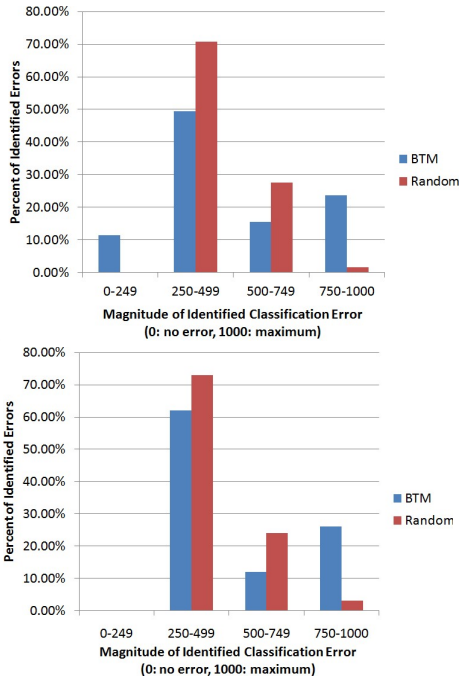

Fig. 3. Visual representation of error severity in model predictions for two tasks related to ad safety: hate speech and adult content detection. The blue bars represent errors identified by the BTM method, while the red bars represent errors identified through random sampling. Each bar corresponds to the percentage of mistakes found in a specific error range.

Are the errors isolated or part of a broader pattern? A key question that arises is whether the errors identified by BTM are isolated incidents or whether they reflect consistent patterns. We approached this question with the assumption that patterns can be modeled, though it's possible that some patterns fall outside the scope of our modeling assumptions.

To explore this further, we conducted an experiment. We tried to train a model that would distinguish between positive and negative examples from the errors identified by BTM. If the errors are consistent and show internal coherence, they are not isolated outliers but represent systematic failures within the model (possibly failures the model is unaware of). The results of this experiment are shown in Figure 6.

Let's start by examining two learning curves: the "btm only" curve and the "student only" curve, which show the 10-fold cross-validated AUC values.

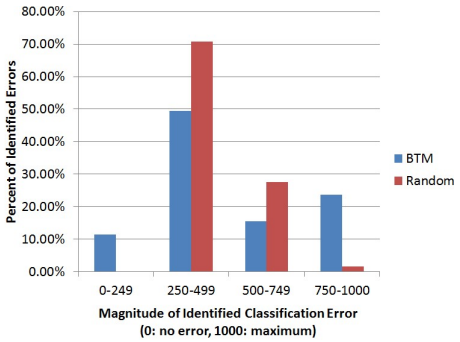

Fig. 4. Error severity distribution for the hate speech classification task, showing mis- takes identified by BTM and random sampling.

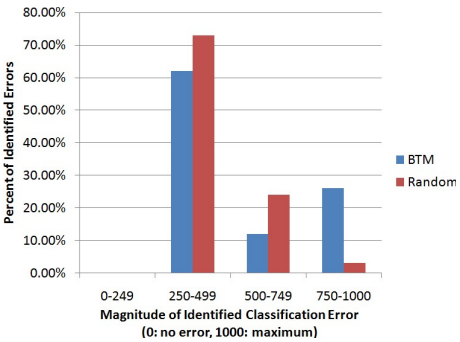

Fig. 5. Error severity distribution for the adult content classification task, with mis- takes identified by BTM and random sampling.

The "btm only" curve represents the performance of models trained and tested with the errors identified by BTM, while the "student only" curve represents the performance of models trained with examples selected through stratified sampling (the student inspection process for these pages gives the "student" label). Notably, both models show high-quality results, indicating that the errors identified by both approaches have internal consistency. The ability of the BTM-trained model to achieve high accuracy suggests that BTM identifies systematic regions of errors that would otherwise go unnoticed, rather than just random outliers.

However, there is a noticeable difference between the quality achieved when training with the two different datasets. The stratified sampling model shows lower accuracy, suggesting that these errors are harder for the model to learn from. This is consistent with our earlier discussion that stratified sampling focuses on uncertain, ambiguous cases, while BTM identifies systematic errors in areas that the model has consistently misclassified.

We also wanted to explore whether the two approaches (DVSRE and BTM) find similar types of errors or if they detect different issues. To test this, we evaluated BTM-trained models on the DVSRE examples and vice versa. The results show that there is little overlap between the two sets of identified errors. BTM findings have little utility for classifying errors identified by DVSRE, and vice versa. This suggests that the two methods identify errors in distinct parts of the model's space—crucially, BTM uncovers errors that are systematically different from those identified by DVSRE.

In conclusion, BTM and DVSRE serve complementary roles in error detection. DVSRE mainly identifies cases where the model is uncertain, while BTM reveals unknown unknowns—errors that the model systematically misses. Even if DVSRE were restricted to identifying only the "unknown" region, it would still fail to uncover as many unknown unknowns as BTM does.

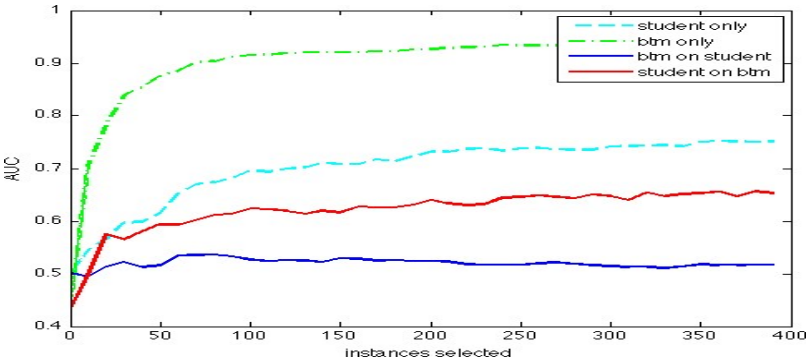

Fig. 6. Learning curves generated from cross-validation (with BTM and student mod- els), followed by testing BTM on errors identified through random sampling (BTM on student errors), and vice versa (students on BTM errors).

6 Influence on Industrial Applications

The Beat the Machine (BTM) approach has reshaped how companies evaluate predictive systems. A mid-sized online advertising firm shifted from traditional model evaluation (cross-validation, expert review, stratified sampling) to integrating BTM for proactive testing. Their lead data scientist highlighted BTM's impact on identifying performance issues, leading to its broader adoption across multiple classification tasks.

In the online labor market, another company incorporated BTM by encouraging users to find misclassified job descriptions. This helped detect emerging job trends early, enhancing algorithm robustness and user satisfaction.

A third company applied BTM to improve image tagging. Human reviewers now challenge existing tags, ensuring suggestions are more accurate and specific, minimizing general or irrelevant labels.

Overall, BTM has transformed model testing by incentivizing the discovery of systematic failures—drawing from security practices but tailored for statistical models.

7 Conclusion

In conclusion, the *Beat the Machine* (BTM) framework introduces a novel, gamified approach to identifying systematic errors in predictive models by leveraging human input. The method not only detects a higher volume of misclassifications than traditional Stratified Random Sampling but also identifies more severe, high-confidence errors that conventional techniques often overlook. Experimental evaluations on classifiers for hate speech and adult content confirm that BTM enhances model robustness and provides critical insights into underlying systematic failures. Moreover, the successful integration of BTM into industrial applications underscores its potential to transform quality assurance processes. By actively engaging users in error detection, the framework bridges the gap between automated evaluation and human insight, leading to a more comprehensive understanding of model limitations.

Future work should focus on refining the incentive mechanisms to further motivate precise error identification and on developing adaptive strategies that adjust rewards based on error severity and occurrence frequency. Additionally, incorporating BTM findings into model retraining pipelines could pave the way for continuous system improvements. Expanding the framework to other domains—such as medical diagnosis or financial forecasting—may also reveal new insights into the generalizability and broader impact of the approach. Overall, BTM represents a significant advancement in addressing unknown unknowns in predictive modeling, and its ongoing development promises to enhance the reliability and interpretability of automated decision systems.

Disclosure of Interest: The author declares that he has no conflict of interest.

References

1. Badami, S.: Optimizing reinforcement learning in partially observable environments using compressed suffix memory algorithm. In: 2024 IEEE 2nd International Conference on Electrical, Automation and Computer Engineering (ICEACE). pp. 330–335 (2024). https://doi.org/10.1109/ICEACE63551.2024.10899025
2. Chow, C.: On optimum recognition error and reject tradeoff. IEEE Transactions on Information Theory **16**(1), 41–46 (1970)

3. Chow, C.K.: An Optimum Character Recognition System Using Decision Func- tions. IEEE Transactions on Electronic Computers **4**, 247–254 (1957)
4. Cochran, W.G.: Sampling Techniques. John Wiley & Sons, New York, 3rd edn. (1977)
5. Domingos, P.: Metacost: A general method for making classifiers cost-sensitive. In: KDD '99 (1999)
6. Elkan, C.: The foundations of cost-sensitive learning. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI). pp. 973–978 (2001)
7. Elkan, C.: The Foundations of Cost-Sensitive Learning. In: IJCAI. pp. 973–978 (2001), http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.29.514
8. Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: SIGIR '94. pp. 3–12. Springer-Verlag New York, Inc., New York, NY, USA (1994). https://doi.org/10.1145/62437.62470, http://dx.doi.org/10.1145/62437.62470
9. Raykar, V., Yu, S., Zhao, L., Jerebko, A., Florin, C., Valadez, G., Bogoni, L., Moy, L.: Supervised Learning from Multiple Experts: Whom to trust when everyone lies a bit. In: ICML. pp. 889–896. ACM (2009)
10. Reiter, R.: On closed world data bases. In: Logic and Data Bases. pp. 55–76 (1977)
11. Settles, B.: Active learning literature survey (2010)
12. Sheng, V.S., Provost, F., Ipeirotis, P.G.: Get another label? improving data quality and data mining using multiple, noisy labelers. In: KDD '08 (2008)
13. Weiss, G.M.: The impact of small disjuncts on classifier learning. In: Stahlbock, R., Crone, S.F., Lessmann, S. (eds.) Data Mining, vol. 8, chap. 9, pp. 193–226. Springer US, Boston, MA (2010). http://dx.doi.org/10.1007/978-1-4419-1280-09