Proposing Threshold Concepts in Machine Learning

Lisa Zhang
lczhang@cs.toronto.edu
University of Toronto Mississauga
Mississauga, Ontario, Canada

Gosia Migut
M.A.Migut@tudelft.nl
Delft University of Technology
Delft, Netherlands

Jesse Hendrik Krijthe
J.H.Krijthe@tudelft.nl
Delft University of Technology
Delft, Netherlands

Abstract

Machine learning (ML) courses are difficult to design and teach for many reasons: new approaches emerge rapidly, and students are already overwhelmed with the volume of material. This position paper addresses the question of what is essential to teach in ML courses through the framework of threshold concepts—transformative ways of understanding that fundamentally shift how learners view a discipline. We propose five threshold concepts in ML: "Learning is Optimization," "There are Tradeoffs in Sources of Error," "ML is an Empirical Science," "ML Describes Geometric Processes," and "ML Demands a Probabilistic Lens." We believe the first three concepts to be essential for students aiming to become adept builders of ML systems, and the remaining two to be necessary for graduate-level research in developing new ML methods. For each concept, we analyze whether and how it is transformative, troublesome, integrative, irreversible, and bounded. These concepts are intentionally chosen to be broad and model-agnostic, so that they can be applied to a wide range of settings and remain relevant as the field progresses. Our proposals reflect the reasoned analysis of experienced ML educators and offer a learner-centered framework for prioritizing limited instructional time while helping students develop a deeper understanding of ML. Furthermore, we contribute to the ongoing conversation about the role of mathematics in ML education by arguing that mathematics is necessary for the two theoretical concepts and for moving beyond mimicry to true understanding through the liminal space of threshold concepts.

CCS Concepts

• **Social and professional topics → Computing education**.

Keywords

threshold concepts, artificial intelligence, AI, machine learning, ML, machine learning education, computing education, math in ML

ACM Reference Format:
Lisa Zhang, Gosia Migut, and Jesse H. Krijthe. 2026. Proposing Threshold Concepts in Machine Learning. In *Proceedings of the 31st ACM Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2026), July 10–15, 2026, Madrid, Spain.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3803400.3809311

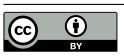
1 Introduction

Machine learning (ML) has become increasingly prominent in computing education. The ACM's 2023 Computer Science Curricular Guidelines include a 40% increase in recommended Artificial Intelligence (AI) and ML instruction hours [26], while the number of university AI programs worldwide has grown from approximately 1,170 in 2018 to 2,520 in 2023 [41]. This curricular expansion raises questions about content prioritization. Students report that the volume of material in ML courses is challenging [38], as ML courses can include foundational mathematics, algorithmic understanding, hands-on implementation, and ethical considerations, yet instructional time remains limited. Efforts such as the Computing Research Association's LEVEL UP AI initiative [13] indicate ongoing work to identify essential content, though consensus on what constitutes core understanding in ML is still developing.

This position paper approaches the question of "what is essential to teach in ML" through the framework of **threshold concepts**, originally proposed in Meyer and Land [30]. Threshold concepts represent both a pedagogical framework and a theory of learning centered on transformative experiences. Threshold concepts are "akin to a portal, opening up a new and previously inaccessible way of thinking about something. It represents a transformed way of understanding, or interpreting, or viewing something without which the learner cannot progress" [30]. Threshold concepts are transformative (fundamentally shift the learner's perspective), troublesome (difficult to grasp and may conflict with prior understanding), integrative (unify disparate concepts in a domain), irreversible (once understood, unlikely to be forgotten or unlearned), and bounded (limited to specific disciplinary boundaries) [30]. In various fields, e.g., writing pedagogy, there is success in using writing threshold concepts [1] to guide first-year composition courses [20]. Attention to threshold concepts can help keep a course focused, while shifting students' perspectives. Although ML courses differ in scope and learning objectives, they can be overwhelming to students in terms of content coverage [38]; understanding threshold concepts can help instructors prioritize limited instructional time and inform curriculum development [5].

Threshold concepts are distinct from other ideas in curriculum design. Bruner [11] emphasized teaching the fundamental structure of a discipline through its "basic concepts", while Wiggins and McTighe [47] describe using "big ideas" with lasting value to guide curriculum construction. Related to these are *core concepts*, the fundamental building blocks or facts within a discipline that students must know [30]. These approaches center on disciplinary structure, while threshold concepts center on the learner's transformative experiences [14, 37]. Unlike concepts that can be learned incrementally, threshold concepts require learners to pass through liminality, a state of "being in between" where understanding may approximate "mimicry or lack of authenticity" [30] as students engage in ritualistic behaviour before achieving mastery [27].

In this position paper, we propose five threshold concepts in ML that bridge theoretical and practical ways of reasoning about



ITiCSE 2026, Madrid, Spain

ACM ISBN 979-8-4007-2634-7/2026/07
https://doi.org/10.1145/3803400.3809311

learning. For each concept, we argue whether and how it is transformative, troublesome, integrative, irreversible, and bounded. The authors are ML educators and researchers with a combined 28 years of ML teaching experience, from North America and Europe. Where possible, we cite related work in ML, computing education, and other areas in our arguments.

2 Background: Threshold Concepts

Threshold concepts have been identified in many fields, including Computer Science [10, 35]. Methods for identifying threshold concepts include consensus-based approaches [5], e.g., consulting with teachers, students, practitioners and community members [33, 42]. These approaches have been used in computing [23] and beyond [5].

Threshold concepts are not without critics. Challenges include lack of a formal definition [33], debates over which of the five characteristics are necessary [49], questions about conceptual granularity [49], and difficulties (or impossibility) in producing empirical evidence [34]. We acknowledge that establishing threshold concepts with empirical rigor may be neither feasible nor necessary for the framework to provide pedagogical value. We contend that carefully reasoned analysis by a few domain experts with deep pedagogical experience offers a meaningful starting point.

Within ML, Allen et al. [2, 3] work towards identifying threshold concepts by considering troublesome concepts, and identify neural networks (including backpropagation) as a potential candidate. We differ from these works in our perspective on what is considered a "concept" (a common disagreement in threshold concept discussions [33]). Moreover, Sibia et al. [38], using student reports of their challenges in a university ML course, argues that the difficulty with neural networks and backpropagation could be due to the way that linear algebra and vectorization are taught. There is work investigating the extent to which math preparation is a difficulty [32], rather than the liminality of the idea itself. Like the writing pedagogy work of Adler-Kassner and Wardle [1], which suggests threshold concepts like "Writing Is a Social and Rhetorical Activity", our proposed concepts are more broad. We aim for concepts that are not specific to particular ML models, but that are *integrative* in the sense that they bring together ideas from across the entire field.

A similarly broad approach is taken in K12 AI Education, where Touretzky et al. [43, 44] identifies 5 "big ideas" in AI, including that computers perceive the world using sensors (perception), agents maintain representations of the world to use for reasoning (representation and reasoning), computers can learn from data (learning), intelligent agents require many kinds of knowledge to interact naturally with humans (natural interaction), and that AI can impact society in both positive and negative ways (societal impact). AI4K12 provides a framework for analyzing and creating instruction in AI literacy for future *users* of AI systems. In contrast, our focus is on addressing what is needed to become *builders* of ML systems—e.g., those who *apply* existing techniques and *research* new ones.

Finally, we note some similarities and differences between our approach and the ACM CS2023 guidelines for ML topic coverage [15, 26]. Many of the suggested topics overlap with our proposed threshold concepts: "evaluation", "application", and "ethics" are all aspects of the empirical nature of ML. In Kumar and Raj [26], the "Fundamental ideas" covered include No Free Lunch and Bias-Variance decomposition, which we also emphasize. However, while ACM CS2023 necessarily provides guidance on topics to cover (e.g., reinforcement learning), we do not. We intentionally chose a broad formulation of a "concept", in an attempt to future-proof, by prioritizing enduring ideas over specific models.

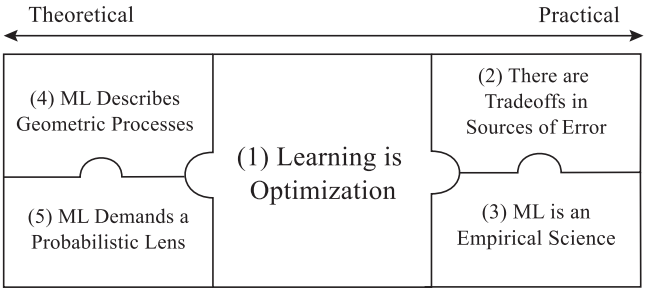


Figure 1: Proposed machine learning threshold concepts, ranging from theoretical (left) to practical (right). The jigsaw shape shows how these concepts are interrelated.

3 Proposed ML Threshold Concepts

Figure 1 shows how the 5 proposed threshold concepts are interrelated. We believe that the three threshold concepts on the right (1-3) are required for *applying* existing ML tools with a working understanding of how ML models behave. The remaining concepts (4-5) are more theoretical, and required for graduate-level work in building new ML approaches.

3.1 Learning is Optimization

The term "learning" evokes imagery of a complex psychological process [29]. In reality, learning problems are almost always turned into optimization problems. Most approaches to learning from data can be described as formulating an objective function (based on a loss function) and finding a function (predictor) that minimizes this objective over some data set. A combination of choices for the *search space* (i.e., model class), choice of *loss function*, and *optimization algorithm* define an ML method. As such, the abstract idea of "learning" is turned into a concrete mathematical procedure.

We consider this idea to be **troublesome**, as it is counter to novices' beliefs about AI/ML systems. Novices tend to anthropomorphize AI/ML systems [8] and believe that ML systems work similarly to the human brain [4, 28, 29]. Novices also hold beliefs that ML systems' behaviour is programmed [29] or that training data is stored [8]. Multiple learning theories suggest that knowledge which conflicts with prior beliefs is difficult to integrate [6, 31].

However, the idea is **transformative**. We believe that it demystifies "learning" by turning it from an intuitive notion to a formal and mechanical procedure. This conceptualization also helps to define the goals of an ML method. This shift allows researchers to construct and communicate new ML methods by specifying the optimization problem. For example, Goodfellow et al. [17] introduces the Generative Adversarial Network (GAN) with the objective

$$\min_G \max_D \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$$

along with the optimization algorithms for the generator G and discriminator D. In this case, it is understood that D and G are neural networks and that a gradient-based optimization approach is used, but other architectures are possible. Turning a learning problem into an optimization problem exposes the fundamentally modular approach to ML, where data, model, and loss form axes along which methods can differ [21].

This shift also has implications for ML model design: the behaviour of an ML model can be influenced by intentionally modifying the objective function. For example, in *regularization*, an extra term is added to the objective function to encode a preference for simpler models. In a classification problem, preference for one class over another can be encoded by adding class weights to the logistic loss. Fairness regularizers [24] are yet another example of the potential that can be unlocked by modifying the objective function.

This concept offers an **integrative** view of ML, describing supervised, unsupervised, and reinforcement learning. This concept also allows practitioners to understand new methods they encounter by asking "what is this process implicitly optimizing?" For example, in the unsupervised learning algorithm k-means clustering, datasets are grouped into k clusters. The algorithm iteratively assigns data points to the nearest cluster centroid (mean), and recalculates the centroids based on the current assignments. Although this description is procedural, this process can be viewed as block coordinate descent on the sum of squares between each data point and its assigned centroid.

This idea is fairly **bounded**. While optimization is used in other areas of CS, the specific conceptualization of "learning as optimization" is distinctive to ML. In other CS domains, optimization is typically used to solve well-defined problems (e.g., finding shortest paths, scheduling tasks), where the objective function is given and the goal is to find optimal solutions. In ML (and sometimes in AI), the optimization perspective reframes the notion of "learning from data" as *constructing* and minimizing an objective function, so the choice of objective becomes a design decision.

We believe the **irreversibility** of this and other proposed threshold concepts to be more complicated. For each concept, we generally believe the conceptual transformations to be irreversible, but it is difficult to provide an argument aside from our experience teaching ML. For example, for this concept, once a student has integrated the perspective that learning methods can be decomposed into their constituent parts, some of the mystery surrounding ML is removed. After viewing optimization as a fundamental tool for understanding ML, we believe that it is difficult to "unsee" this perspective when analyzing new methods or debugging models.

However, we do not believe consistent practical application of many of these concepts to be irreversible. For this concept, students may forget to apply the optimization perspective in practice, instead remembering only the procedural rituals (e.g., train-test split, hyperparameter tuning) that emanate from this conceptualization. Moreover, as new methods emerge, students may return to having a mystical view: for example, "prompting" a Generative AI model can be seen as abstracted away from the optimization process that trained the model, potentially obscuring the connection to the "learning is optimization" perspective. Nevertheless, once the optimization lens has been adopted, it remains available as a tool for understanding.

3.2 There are Tradeoffs in Sources of Error

While novices can hold prior beliefs that ML systems are precise [29] and perfect [8], ML models make mistakes, and there are tradeoffs involved in managing those mistakes. The Bias-Variance tradeoff is the most commonly discussed tradeoff, emphasized in many ML textbooks [9, 18, 19]. The Bias-Variance decomposition shows that there are three sources of error: *Bias* (due to a model being "too simple" or having the wrong "inductive bias"), *Variance* (due to a model that is too complex, capturing not only the salient but also the accidental patterns in the training data), and the *Bayes error* (resulting from natural variations in the target variable not associated with the input features). This decomposition guides the tradeoff in how model complexity is managed.

We intentionally worded this threshold concept to consider other tradeoffs. This is partly because a direct application of the Bias-Variance decomposition is nuanced [7], and partly because there are other tradeoffs involved in ML practice, e.g., in the data representation and computational resource use. For example, although Bayes error is sometimes called the *irreducible error* (as no model can achieve a lower error on the given problem), it is tied to choices in the problem formulation, and is affected by the representation of the input data. This highlights a fundamental aspect of ML: poor-quality or unrepresentative data inevitably lead to poor models. However, overly rich data representation can lead to a model that is too complex for the amount of training data, resulting in high variance, reflecting a tradeoff. Thus, many limitations of ML systems cannot be solved through better algorithms alone, and there are tradeoffs involved in choices regarding the model *and* the data.

That there are tradeoffs in errors is a **troublesome** idea in that it has several counterintuitive aspects. First, unlike what a novice might believe, a more complex model is not always better, and a higher training accuracy does not always mean a "better" model. Second, the fact that ML algorithms are not magical thinking machines but computational procedures that depend on the patterns present in their training data is counter to students' prior beliefs [8]. Many limitations of ML systems cannot be solved through better algorithms alone, because the root cause often lies in poor-quality or unrepresentative data. In addition, the mathematics behind the Bias-Variance decomposition is itself challenging. The mathematical derivation is typically shown for regression models, and it takes mathematical maturity to abstract this specific Bias-Variance decomposition and intuit how it can generalize to other scenarios.

This idea is **integrative** in that the Bias-Variance tradeoff and other tradeoffs apply to many models from across ML. Many different ML models have at least one hyperparameter controlling the complexity of the model, e.g., k in k-nearest neighbours, maximum depth in decision trees, number of layers in a neural network. In all of these models, the representation of the data is yet another key factor contributing to the error of the model. These tradeoffs tie together various concepts related to learning curves, cross-validation, and hyperparameter tuning.

The idea is **transformative** in that it helps a student move beyond trial and error for model improvement, by providing a conceptual framework to understand how different aspects of model and data lead to different tradeoffs, and how to move towards an optimum. In terms of practical application, it helps a student learn

how to debug appropriately in applied ML settings. A novice may try many different model settings to see what works, but an expert will analyze the sources of error and adjust the model complexity and data choice accordingly.

The idea is **bounded**: within computer science, it is specific to machine learning and statistical learning. The Bias-Variance tradeoff specifically is usually formalized in supervised learning contexts, but can be applied conceptually to unsupervised learning and reinforcement learning.

As in the previous concept, while we believe conceptual understanding to be **irreversible**, the applications of the idea are challenging. For example, Valdenegro-Toro and Sabatelli [46] show that students hold many beliefs about overfitting.

3.3 Machine Learning is an Empirical Science

A closely linked idea in ML is the understanding that no single algorithm is universally "best". The "No Free Lunch" [48] theorem provides the theoretical foundation for this insight, stating that "any two optimization algorithms are equivalent when their performance is averaged across all possible problems." Already, this idea contains difficult notions like "averaging across problems". In short, the theorem implies that there is no universally best algorithm; success is always context-dependent. Performance depends on how well the data, model, and loss function align for a specific problem, and this alignment cannot be achieved using theory alone.

We therefore rely on empirical evaluation using observed data when building ML solutions and assessing their performance. This evaluation includes most aspects of ML model building, such as model selection and hyperparameter tuning. Empirical evaluation means assessing a model's performance using observed data. This includes analyzing the error decompositions, for example by plotting learning curves that compare training and validation loss over training iterations to understand model generalization and identify signs of overfitting (and underfitting).

The realization of the importance of empirical evaluation **transforms** how learners think about machine learning. They move from a "best algorithm" mindset (of trying to find the best algorithm to study) to a problem–data–model alignment mindset: which combination of model, loss, and data is most appropriate for the given task and goals? This shift from algorithm-centric thinking to alignment thinking is the beginning of understanding ML as an empirical science. It places proper evaluation at the center of ML practice, rather than an interesting aside.

However, loss minimization is only one aspect of empirical evaluation. This realization represents a second transformative shift from single-metric thinking to multidimensional, evidence-based, human-centered reasoning. For example, a model that predicts cancer risks may have equal accuracy across men and women, but may have different distributions of false positives vs. false negatives. A facial recognition system may perform well across the selected validation/test sets, but have significantly higher error rates for darker-skinned individuals [12]. A model might perform well in detecting pneumonia from chest x-rays, but only because it has learned to leverage hospital-specific visual markers [50]. Thus, for a model to perform "well" in practice, it should not only be accurate, but also efficient, robust, fair, and interpretable—and many of these attributes conflict with one another. Thus, empirical evaluation also involves understanding how, why and for whom the model works, and what procedures need to be in place if models fail, including systematic auditing to identify potential issues before deployment.

The central role of empirical evaluation is **troublesome** in many ways. First, we do not believe that the transformative shifts above happen instantaneously. In our experience, some students find it difficult to accept the inadequacy of theory to determine the "best" model, and find the empirical approach unsatisfying—as if requiring a mourning period. In statistical learning, decisions are often made by making assumptions about the data distribution, and considering asymptotic or expected behaviours of various choices. However, in practical applications, these assumptions rarely hold. Thus, while simplifying assumptions can still be useful, theoretical guarantees offer little comfort in safety-critical systems like those used in self-driving cars, and need to be supported by empirical evaluation.

Second, consistently performing good empirical evaluation is troublesome. For example, Skripchuk et al. [40] and Zimmermann et al. [51] both show that students underestimate the influence of hyperparameters. This underestimation is not limited to students: hyperparameter tuning is often underutilized or improperly conducted in ML research [39], even leading to cases where well-tuned baseline methods outperform more sophisticated approaches that have not been properly tuned [22]. This demonstrates that evaluation is both difficult and essential to building ML solutions.

Empirical evaluation is **integrative** because it provides a single lens through which to interpret many failure modes: overfitting and underfitting, sensitivity to hyperparameter choices, poor generalization, bias in training data, instability across different datasets due to various types of distribution shifts, and data leakage (e.g., from the training to the test data). The broad phrasing of this concept takes the stance that issues of unintentional harm (i.e., risk of malfunction) caused by the use of ML systems are firmly within the bounds of our discipline.

As with previous concepts, while conceptual understanding may be **irreversible**, it is easy to slip back into treating models as inscrutable, especially when new, large models seem to be able to do well on many learning problems. Moreover, as we have seen, consistently applying best-practices in data splitting, model selection, and hyperparameter tuning is difficult.

It is **bounded**, in that it delineates ML from other parts of CS, where theoretical optimality claims are more common, and empirical evaluation plays a less central role. This concept reframes machine learning as an empirical, data-dependent discipline grounded in alignment and evidence. It unites understanding of bias and variance, generalization, and model selection under a single principle: that reliable knowledge in machine learning comes not from theoretical optimality, but from careful, multidimensional empirical evaluation of how data, algorithms, and models interact.

3.4 ML Describes Geometric Processes

The remaining two threshold concepts are more theoretical in nature. As before, there are multiple components to this idea. At the most basic level, data can be viewed as a *geometric object* (i.e., points in $\mathbb{R}^D$), and models as *geometric processes* acting on these objects. Of course, the choice of data representation matters; different choices

lead to differences in distances between data points, and the symmetries, invariances, and equivariances afforded. This perspective frames both supervised and unsupervised learning models as transformations on the data space that warp the representation space. At a more advanced level, models themselves can also be thought of as geometric objects: distance, symmetry, and other geometric concepts can be applied to models as objects. When conceived in this manner, the objective function and optimization processes can be thought of and visualized as geometric processes that navigate through the model space.

This idea is **troublesome**. Thinking of data as points in $\mathbb{R}^D$ is so "obvious" that many ML textbooks gloss over this idea, but it is counter to intuition developed in prior CS courses. Earlier CS courses often emphasize differences in data types, with text represented as a sequence of characters, images as an array of pixels, and sound as waveforms sampled at regular intervals. Data structures are usually analyzed through the lens of time and space complexity of typical operations. The geometry of the representation—such as distances between data points, relationships in high-dimensional space, and symmetries—is rarely considered.

Additionally, in our experience, students struggle with ideas and language like "the earlier layers of a Multi-Layer Perceptron learn features, and the final layer is a linear classifier on these features" and "natural images lie in a manifold within $\mathbb{R}^D$"; these ideas that ML practitioners take for granted need to be broken down for novices. Empirically, Sibia et al. [38] shows that students find "visualization" or "intuition" behind mathematical ideas in ML challenging, with quotes from students suggesting difficulty connecting the algebraic presentation of ML and the geometric. Even for experts, reasoning about geometry in high-dimensional spaces is difficult, as intuitions in low-dimensional spaces do not always translate.

The geometric perspective is **integrative**. It provides a unified perspective on how data, models, and loss interact, and how different models and algorithms can be combined. The geometry of the data space influences what predictors can learn (e.g., why categorical variables need to be one-hot encoded). The geometry of the model determines what predictors are possible (e.g., inspiring models like Convolutional Neural Networks and Graph Neural Networks that work with the symmetries and equivariances in the data) and provides interpretations of what the model is doing (e.g., normalizing flow, diffusion models). The geometry of the loss landscape determines how we can find good predictors (which helps explain gradient descent and interpret its variants, e.g., momentum, adaptive step-sizes, gradient clipping, etc.). This perspective is thus used throughout advanced texts and discourse.

These examples show that this idea is **transformative**. Once it is internalized, it not only unlocks understanding of the principles behind many models, but also enables one to go beyond analyzing specific data sets and their features. Thinking geometrically (in terms of distances, symmetries, and invariances) unlocks both simple ideas (like one-hot encodings) and advanced ideas (like normalizing flows and diffusion models). Ideas like the way embeddings work are useful not only in theory, but also in practical application, including those involving LLMs.

As with other ideas, we believe the conceptual approach to be **irreversible**. It remains challenging to apply these concepts consistently in practice, even for experts.

While geometric ideas are not **bounded** to ML, the way geometry is used in ML is distinctive compared to other areas of CS. For example, computational geometry (where algorithms operate on geometric objects like polygons and point sets), computer graphics (where 3D transformations and projections are fundamental), and algorithm design (where geometric data structures like KD-trees and spatial hashing are used for efficient queries) all use geometric ideas. However, the emphasis is typically on the discrete structure and relationships between nodes in a graph-based space. ML emphasizes continuous, high-dimensional spaces, and the geometry of how models act on them.

3.5 ML Demands a Probabilistic Lens

A related concept in ML is understanding that deterministic reasoning is insufficient, that data, models, predictions, and even optimizers are best approached through a probabilistic lens. At a basic level, this means understanding that data are samples drawn from the underlying probability distributions, and the learning process involves reasoning about what is likely rather than what is certain. The predictions are estimates of properties of distributions rather than single outcomes, generalization is reasoning about expected performance, and interpretability is not just about explaining outputs, but about connecting a model's representations to the probability distributions it assumes or learns from.

This idea is **troublesome**. Prior work suggests that novices struggle with the probabilistic aspect of ML [25, 29]. Sibia et al. [38] shows that undergraduate ML students report "probability and statistics" to be challenging. We observe, in our own courses, that CS students struggle with probabilistic reasoning; deterministic reasoning is much more common in other areas of CS. We also observe that students can go a long way in ML without understanding probability. Within the liminal space, they can train a classifier, tune hyperparameters, evaluate accuracy, and deploy a predictive model by following ritualistic formulas. It is thus possible to apply ML without probability, just as one can drive a car without understanding how the engine works. However, novices will not be able to reason deeply about ML without probabilistic thinking.

Probability is not just a branch of math in ML; it fundamentally **transforms** the view of learning as a stochastic phenomenon, rather than a deterministic algorithm. Randomness not only shows up in how training data are collected but is also intentionally used in the design of algorithms. Many ML models and loss functions have probabilistic interpretations. Some models make these probabilistic interpretations and assumptions explicit: the GAN objective, as shown in Section 3.1, is written in terms of expectations over p_{data} and p_z. Others (like k-means) hide core assumptions behind seemingly deterministic procedures. Being able to understand and use these assumptions is key to understanding the difference between an autoencoder and a Variational Autoencoder, including the reparameterization trick. Advanced ideas like reasoning about expectations over actions and states in Reinforcement Learning all require probabilistic reasoning. Advanced ML models are often reasoned about through their probabilistic interpretation of

data and its generating processes, in order to build better models. Bootstrapping and forms of regularization all rely on it, intentionally introducing randomness; thus, stochasticity connects different aspects of ML.

One way that this transformation can be observed is whether students are able to see randomness not just as an obstacle, but as a tool. For example, data augmentation is a strategy where noise is added to the training set to mimic the variability we expect in real-world data, to build models that are more robust to this variability. Beyond data augmentation, randomness is also deliberately introduced in *differential privacy*, where it becomes a design principle, rather than a flaw, used to enrich data, protect privacy, and enhance generalization. Another more basic example is Stochastic Gradient Descent (SGD), where randomness in mini-batch selection helps escape shallow minima.

As these examples come from many areas of ML, this idea is **integrative**. It combines multiple perspectives on why learning can be hard or impossible, including Bayes error, the curse of dimensionality, and fairness notions, that all illustrate ways that uncertainty and probabilistic reasoning shape the limits of learning.

Unlike previous concepts, probabilistic reasoning under uncertainty is **not bounded** to the field of ML. It connects closely to statistics, a related field, and is used in other fields that model phenomena, like physics, econometrics and finance. Moreover, many areas of CS (e.g., algorithm design, operating systems) do use probabilistic reasoning to analyze, design, and improve algorithms. However, some CS disciplines view non-determinism as a problem. In ML, when probabilistic reasoning is understood, it is used purposefully to prevent overfitting, improve robustness, and make algorithms more efficient, adding to the transformative perspective on stochasticity. Thus, taking a probabilistic perspective means that noise is not always a problem, and can sometimes be a solution. However, this is true in other fields that take a probabilistic perspective.

As before, we believe the theoretical ideas to be **irreversible**, but our experience suggests that in practical application, students sometimes lose this probabilistic lens.

4 Implications for ML Pedagogy

As new ML models and methods are constantly introduced, we believe that a small number of model-agnostic ideas should guide curriculum design. We intentionally chose concepts that are "larger" in granularity than individual models, with the goal of proposing threshold concepts that can remain relevant over time.

Implications for Existing ML Courses. Even without fully redesigning courses, ML instructors can use these threshold concepts as recurrent threads to tie together different models. As of the writing of this paper, many ML textbooks and courses (including ours) use a "survey of techniques" approach, where a new ML model is introduced in each unit. This works well for a reference text. However, novices can fail to understand what is important, foundational, and cross-cutting when little connection is made between chapters. Referring back to the threshold concepts can help prompt important transformations to occur.

For New Courses and Curricula. For a course or curriculum redesign, these threshold concepts can provide a guide for what is important, in a way that keeps a learner's perspective in mind. Not all threshold concepts need to be emphasized in all ML courses. For an upper-year undergraduate or graduate ML course, it is possible to introduce the ideas from "middle then outwards" in Figure 1. For a more basic ML course, it is possible to start with the practical ideas on the right, treating the models as opaque tools; application-based ML courses for majors and non-majors alike have been described in the literature [16, 45].

Overall, which threshold concepts to introduce depends on the goals of the course and the students it serves. We propose that instructors should first consider what role they want students to play (e.g., builders of ML tools, researchers developing new methods, or informed users of ML systems), then identify what transformations students should experience to fulfill that role, and finally build (and rebuild) the course topics and sequence as necessary.

Usefulness of Mathematics. One recurring question in ML education is the extent to which mathematics is necessary in ML [36]. An approach to answering this question is to consider the transformations that learners should achieve. For the applied threshold concepts, we believe that it is possible for learners to pass the liminal space with minimal background in calculus, linear algebra, and probability. More specifically, understanding that ML is an empirical science and the multifaceted nature of model evaluation can be done without math. It is also possible (though to a lesser extent) to intuitively understand the sources of error and key tradeoffs.

For the remaining concepts, mathematical maturity and prerequisites can be an important *pre-liminal variation* [30, 33] to consider. Without the mathematics, we believe that learners are less effective at "negotiating the liminality" [33], and transitioning from mimicry to understanding. Learners will miss key insights related to the geometry of the problem, the probabilistic nature of the data, model, loss, and various aspects of the optimization process. Still, it may be important to teach students without deep background in mathematics, as math (and math self-efficacy) remains a barrier for ML learners. With tailored instruction, we believe learners can still go far, but the transformations may not be irreversible. Careful instruction should consider how to mitigate the reversibility.

5 Conclusion

As ML continues to evolve rapidly, identifying stable, foundational ways of thinking about the field becomes increasingly valuable for both educators and learners. These five proposed threshold concepts can stimulate discussion, guide curriculum development, and ultimately help students develop a deeper, more transformative understanding of ML. While we hope this work serves as a starting point for building broader consensus on threshold concepts in ML, we believe these proposals offer immediate value for ML educators. Future work could also explore how these concepts manifest in student learning, investigate what additional threshold concepts are important in ML, and examine how different instructional approaches can help students cross these conceptual thresholds.

Acknowledgments

We thank many colleagues for helpful discussions, including Jonathan Calver, Michelle Troberg, Alice Gao, Rahul G. Krishnan, Dan Zingaro, and UTM's Threshold Concept Working Group.

References

[1] Linda Adler-Kassner and Elizabeth Wardle. 2015. *Naming what we know: Threshold concepts of writing studies.* University Press of Colorado.

[2] Becky Allen, Marie Devlin, and A. Stephen McGough. 2021. Using the One Minute Paper to Gain Insight into Potential Threshold Concepts in Artificial Intelligence Courses. In *Proceedings of the 5th Conference on Computing Education Practice*. Association for Computing Machinery, New York, NY, USA, 21–24.

[3] Becky Allen, Andrew Stephen McGough, and Marie Devlin. 2022. Toward a Framework for Teaching Artificial Intelligence to a Higher Education Audience. *ACM Transactions on Computing Education* 22, 2 (June 2022), 1–29.

[4] Pavlo Antonenko and Brian Abramowitz. 2023. In-service teachers' (mis)conceptions of artificial intelligence in K-12 science education. *Journal of Research on Technology in Education* 55, 1 (2023), 64–78.

[5] Sarah Barradell. 2013. The identification of threshold concepts: a review of theoretical complexities and methodological challenges. *Higher Education* 65 (2013), 265–276.

[6] Frederic C. Bartlett. 1932. *Remembering: A Study in Experimental and Social Psychology*. Cambridge University Press, Cambridge.

[7] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. 2019. Reconciling Modern Machine-Learning Practice and the Classical Bias–Variance Trade-Off. *Proceedings of the National Academy of Sciences* 116, 32 (2019), 15849–15854.

[8] Arne Bewersdorff, Xiaoming Zhai, Jessica Roberts, and Claudia Nerdel. 2023. Myths, mis-and preconceptions of artificial intelligence: A review of the literature. *Computers and Education: Artificial Intelligence* 4 (2023), 100143.

[9] Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer, New York.

[10] Jonas Boustedt, Anna Eckerdal, Robert McCartney, Jan Erik Moström, Mark Ratcliffe, Kate Sanders, and Carol Zander. 2007. Threshold concepts in computer science: do they exist and are they useful? *ACM Sigcse Bulletin* 39, 1 (2007).

[11] Jerome S. Bruner. 1960. *The process of education*. Harvard University Press, Cambridge, MA.

[12] Joy Buolamwini and Timnit Gebru. 2018. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*. PMLR, 77–91.

[13] Computing Research Association. 2024. CRA Launches LEVEL UP AI Initiative to Increase Capacity and Inclusion in AI Education. https://cra.org/cra-launches-level-up-ai-initiative-to-increase-capacity-and-inclusion-in-ai-education/. Accessed: 2025-12-01.

[14] Glynis Cousin. 2006. An introduction to threshold concepts. *Planet* 17, 1 (2006).

[15] Eric Eaton and Susan L. Epstein. 2024. Artificial intelligence in the CS2023 undergraduate computer science curriculum: Rationale and challenges. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 23078–23083.

[16] Rebecca Fiebrink. 2019. Machine learning education for artists, musicians, and other creative practitioners. *ACM Transactions on Computing Education (TOCE)* 19, 4 (2019), 1–32.

[17] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems* 27 (2014).

[18] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning* (second ed.). Springer.

[19] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An Introduction to Statistical Learning*. Vol. 112. Springer.

[20] Kristine Johnson. 2019. The threshold concepts of writing studies in the writing methods course. *Teaching/Writing: The Journal of Writing Teacher Education* 6, 1 (2019), 2.

[21] Alexander Jung. 2022. *Machine learning: the basics*. Springer Nature.

[22] Rudolf Kadlec, Ondrej Bajgar, and Jan Kleindienst. 2017. Knowledge Base Completion: Baselines Strike Back. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*. Association for Computational Linguistics, 69–74.

[23] Maria Kallia and Sue Sentance. 2017. Computing teachers' perspectives on threshold concepts: Functions and procedural abstraction. In *Proceedings of the 12th workshop on primary and secondary computing education*. 15–24.

[24] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. 2012. Fairness-Aware Classifier with Prejudice Remover Regularizer. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II*. Springer, 35–50.

[25] Magnus Høholt Kaspersen, Karl-Emil Kjær Bilstrup, Maarten Van Mechelen, Arthur Hjort, Niels Olof Bouvin, and Marianne Graves Petersen. 2022. High school students exploring machine learning and its societal implications: Opportunities and challenges. *International Journal of Child-Computer Interaction* 34 (2022).

[26] Amruth N. Kumar and Rajendra K. Raj. 2023. Computer Science Curricula 2023 (CS2023) Community Engagement by the ACM/IEEE-CS/AAAI Joint Task Force. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 2*. 1212–1213.

[27] Ray Land, Glynis Cousin, Jan H F Meyer, and Peter Davies. 2005. Threshold concepts and troublesome knowledge (3): implications for course design and evaluation. In *Improving student learning: Diversity and inclusivity*, Chris Rust (Ed.). Oxford Centre for Staff and Learning Development, Oxford, 53–64.

[28] Annabel Lindner and Marc Berges. 2020. Can you explain AI to me? Teachers' pre-concepts about Artificial Intelligence. In *2020 IEEE Frontiers in education conference (FIE)*. IEEE, 1–9.

[29] Erik Marx, Clemens Witt, and Thiemo Leonhardt. 2024. Identifying Secondary School Students' Misconceptions about Machine Learning: An Interview Study. In *Proceedings of the 19th WiPSCE Conference on Primary and Secondary Computing Education Research*.

[30] Jan Meyer and Ray Land. 2003. Threshold concepts and troublesome knowledge: Linkages to ways of thinking and practising within the disciplines. (2003).

[31] Jean Piaget. 1952. *The Origins of Intelligence in Children*. International Universities Press, New York.

[32] Ilinca Rențea. 2025. The Role of Mathematics Proficiency in Learning & Teaching ML. In *Proceedings of the 30th ACM Conference on Innovation and Technology in Computer Science Education V. 2* (Nijmegen, Netherlands). Association for Computing Machinery, New York, NY, USA, 828–829.

[33] Janet Rountree and Nathan Rountree. 2009. Issues regarding threshold concepts in computer science. In *Proceedings of the Eleventh Australasian Conference on Computing Education - Volume 95* (Wellington, New Zealand). Australian Computer Society, Inc., AUS, 139–146.

[34] Darrell Patrick Rowbottom. 2007. Demystifying threshold concepts. *Journal of Philosophy of Education* 41, 2 (2007), 263–270.

[35] Kate Sanders and Robert McCartney. 2016. Threshold concepts in computing: past, present, and future. In *Proceedings of the 16th Koli Calling international conference on computing education research*. 91–100.

[36] R. Benjamin Shapiro and Rebecca Fiebrink. 2019. Introduction to the special section: Launching an agenda for research on learning machine learning. *ACM Transactions on Computing Education (TOCE)* 19, 4 (2019), 1–6.

[37] Dermot Shinners-Kennedy. 2016. How not to identify threshold concepts. In *Threshold Concepts in Practice*, Ray Land, Jan H. F. Meyer, and Michael T. Flanagan (Eds.). Sense Publishers, 253–267.

[38] Naaz Sibia, Amber Richardson, Alice Gao, Andrew Petersen, and Lisa Zhang. 2025. Student Perspectives on the Challenges in Machine Learning. In *Proceedings of the 30th ACM Conference on Innovation and Technology in Computer Science Education V. 1*. 9–15.

[39] Sebastian Simon, Nikolay Kolyada, Christopher Akiki, Martin Potthast, Benno Stein, and Norbert Siegmund. 2023. Exploring Hyperparameter Usage and Tuning in Machine Learning Research. In *2023 IEEE/ACM 2nd International Conference on AI Engineering – Software Engineering for AI*. 68–79.

[40] James Skripchuk, Yang Shi, and Thomas Price. 2022. Identifying Common Errors in Open-Ended Machine Learning Projects. In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education - Volume 1*. Association for Computing Machinery, Providence RI USA, 216–222.

[41] Stanford Institute for Human-Centered AI (HAI). 2024. AI Index 2024 Annual Report, Chapter 6: AI Education and Talent. https://aiindex.stanford.edu/report/ Accessed: 2024-10-22.

[42] Julie A. Timmermans and Jan H. F. Meyer. 2019. A framework for working with university teachers to create and embed 'Integrated Threshold Concept Knowledge' (ITCK) in their practice. *International Journal for Academic Development* 24, 4 (2019), 354–368.

[43] David Touretzky, Christina Gardner-McCune, Fred Martin, and Deborah Seehorn. 2019. Envisioning AI for K-12: What should every child know about AI?. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 9795–9799.

[44] David Touretzky, Christina Gardner-McCune, and Deborah Seehorn. 2023. Machine learning and the five big ideas in AI. *International journal of artificial intelligence in education* 33, 2 (2023), 233–266.

[45] Tiffany Tseng, Matt J Davidson, Luis Morales-Navarro, Jennifer King Chen, Victoria Delaney, Mark Leibowitz, Jazbo Beason, and R. Benjamin Shapiro. 2024. Co-ML: Collaborative machine learning model building for developing dataset design practices. *ACM Transactions on Computing Education* 24, 2 (2024), 1–37.

[46] Matias Valdenegro-Toro and Matthia Sabatelli. 2023. Machine learning students overfit to overfitting. In *The Third Teaching Machine Learning and Artificial Intelligence Workshop*. PMLR, 46–51.

[47] Grant Wiggins and Jay McTighe. 2005. *Understanding by design* (2nd ed.). Association for Supervision and Curriculum Development, Alexandria, VA.

[48] David H. Wolpert. 1996. The lack of a priori distinctions between learning algorithms. *Neural computation* 8, 7 (1996), 1341–1390.

[49] Lucy Yeomans, Steffen Zschaler, and Kelly Coate. 2019. Transformative and troublesome? Students' and professional programmers' perspectives on difficult concepts in programming. *ACM Transactions on Computing Education* 19, 3 (2019), 1–27.

[50] John R. Zech, Marcus A. Badgeley, Manway Liu, Anthony B. Costa, Joseph J. Titano, and Eric Karl Oermann. 2018. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLoS medicine* 15, 11 (2018), e1002683.

[51] Renato Magela Zimmermann, Sonya Allin, and Lisa Zhang. 2023. Common Errors in Machine Learning Projects: A Second Look. In *Proceedings of the 23rd Koli Calling International Conference on Computing Education Research*. 1–12.