2026 ASEE ANNUAL CONFERENCE June 21–24, 2026 | Charlotte Convention Center, NC
Host Sponsor UNC Charlotte
Paper ID #52634

Designing and Evaluating an Application-Focused Machine Learning Course for Engineering Technology Students

Dr. Wenhai Li, State University of New York, Farmingdale State College

Assistant Professor in Department of Mechanical Engineering Technology, Farmingdale State College, Farmingdale, NY 11735

Dr. Yue Hung, Farmingdale State College

Dr. Yue (Jeff) Hung holds degrees in engineering and technology disciplines (Ph.D. in Materials Science and Engineering, M.S in Mechanical Engineering, and B.S in Manufacturing Engineering Technology). He has over 20 yearsâ€™ experience in Computer-Aided

Dr. Zhou Zhang, State University of New York, College of Technology at Farmingdale

I am an Assistant Professor at SUNY Farmingdale State College. My teaching and research interests include robotics and virtual reality in engineering education. I have a Ph.D. and a bachelor's degree in Mechanical Engineering, and my master's degree is in Electrical Engineering. I have over seven years of industrial experience as an electrical and mechanical engineer. I also have extensive teaching and research experience with respect to various interdisciplinary topics involving Mechanical Engineering, Electrical Engineering, and Computer Science.

Designing and Evaluating an Application-Focused Machine Learning Course for Engineering Technology Students

Wenhai Li, Jeff Hung, Zhou Zhang
Department of Mechanical Engineering Technology, Farmingdale State College, Farmingdale, NY 11735

Ning Yu
Department of Computer Science, State University of New York at Brockport, Brockport, NY 14420

Foluso Ladeinde
Department of Mechanical Engineering, Stony Brook University, Stony Brook, NY 11794

Abstract

Machine learning (ML) is increasingly used in engineering practice, yet many Mechanical Engineering Technology (MET) programs face challenges in introducing ML due to students' limited backgrounds in mathematics and programming and the constraints of already dense curricula. To address these challenges, a new application-focused course, Machine Learning for Applied Engineering (MET309), was developed and implemented within the MET program at Farmingdale State College. The course emphasizes conceptual understanding, engineering relevance, and hands-on application through staged laboratory activities and projects, rather than mathematical derivation or extensive coding from scratch. This paper describes the course design, instructional methods, and assessment approach, and presents results from the initial offering of the course in Fall 2025. Assessment results indicate that the course design was effective in supporting the introduction of machine learning within an MET curriculum and offers a course design framework that can be adapted within similar engineering technology programs.

1. Introduction

Artificial intelligence (AI) and machine learning (ML) are rapidly reshaping modern engineering practice, enabling new capabilities in automation, data-driven decision making, system optimization, and intelligent manufacturing [1][2]. Across mechanical and manufacturing domains, ML techniques are increasingly embedded within engineering workflows to support tasks such as defect detection, predictive maintenance, adaptive control, and process monitoring [3][4]. As these technologies mature and become more accessible, the ability to apply ML tools effectively has emerged as an important professional competency for engineering graduates, including those trained in engineering technology programs.

Multiple national and international reports have highlighted the accelerating demand for engineers who can integrate AI and ML into applied engineering environments [5]. Industry surveys consistently indicate strong growth in AI adoption across manufacturing and industrial sectors, coupled with a persistent skills gap between workforce needs and current graduate preparation [6]. Employers increasingly seek engineers who can work with data, select appropriate ML tools, and

integrate intelligent components into physical systems, rather than develop algorithms from first principles. These trends suggest that applied ML literacy is becoming a foundational expectation for engineering technology graduates entering the modern workforce.

In parallel with industry demand, student interest in AI and ML has grown significantly. Mechanical Engineering Technology (MET) students increasingly recognize the relevance of ML to their future careers and frequently express interest in learning how these techniques are applied in real engineering systems. In classroom interactions, students often approach instructors to ask whether machine learning concepts will be incorporated into related technical courses, such as robotics, automation, and manufacturing systems. These interactions suggest a growing expectation among MET students that AI/ML skills should be part of their professional preparation, highlighting a disconnect between student interest and the limited formal opportunities currently available within many MET curricula.

To respond to these educational and workforce pressures, the MET program at Farmingdale State College has begun incorporating machine learning content in a structured and intentional manner. While this effort follows a broader scaffolded philosophy for introducing AI/ML concepts across the MET curriculum, the primary focus is the development of a dedicated course, *Machine Learning for Applied Engineering* (MET309). This course is designed to serve as the foundational entry point for ML learning within the MET program, providing students with the essential concepts, practical workflows, and applied experiences needed to engage with ML in engineering contexts. The broader curricular considerations and complementary instructional efforts in our MET program have been discussed in prior publications by the authors [7-9]; the present paper concentrates exclusively on the design and evaluation of MET309.

The subsequent sections of this paper present the design and evaluation of an application-focused machine learning course developed for MET students. First, the challenges motivating the course development and the corresponding instructional goals are described. The course design philosophy is then outlined, followed by a detailed discussion of the course structure and content. Next, the assessment design used to evaluate student learning and course effectiveness is presented. The paper then summarizes key results and observations from the initial course offering. Finally, the discussion and conclusion reflect on the implications of the findings and offer considerations for engineering technology programs seeking to integrate machine learning into their curricula.

2. Challenges and Goals

The design of an effective machine learning course for MET students begins with understanding the challenges these students encounter when learning ML and then identifying the core ML skills that are most relevant to their future engineering practice. Recognizing these challenges helps clarify realistic instructional goals and guides decisions about which machine learning concepts and techniques should be emphasized in an application-focused course.

2.1 Challenges in Teaching ML to MET Students

Teaching machine learning to MET students involves challenges that differ from those typically encountered in computer science or engineering science programs. While MET students are strong

in applied problem solving and hands-on implementation, their academic preparation and curricular structure introduce constraints that must be addressed deliberately when designing an ML course. The primary challenges identified in our MET program are summarized below:

- Limited background in advanced mathematics: Many machine learning methods are traditionally introduced through concepts such as linear algebra, probability, and optimization, which are not heavily emphasized in most MET curricula. When ML instruction relies on mathematical derivations, students may struggle to follow the material or to see its relevance to engineering applications.

- Limited experience with coding: Although MET students regularly use engineering software tools, many have little prior experience with programming languages such as Python. Machine learning tasks often require working with datasets, modifying scripts, and debugging code, which can be overwhelming without sufficient support and gradual skill development.

- Disciplinary mismatch between ML and MET contexts: Many machine learning concepts are presented in ways that feel more closely associated with computer science than with mechanical or manufacturing engineering. When instruction is not explicitly connected to physical systems, sensors, or engineering processes, MET students may find the material unfamiliar and struggle to see how it relates to their engineering training.

- Curricular constraints within MET programs: MET curricula are typically already packed with required technical and laboratory courses, leaving little flexibility to introduce new subject areas through multiple sequential courses. Unlike computer science programs, where machine learning is often taught through a series of prerequisite-heavy courses, MET programs must introduce ML concepts within a much more limited curricular space. As a result, essential machine learning knowledge must be carefully condensed into a single course, making it challenging to balance foundational understanding, practical application, and accessibility for MET students.

These challenges highlight the need for a machine learning course designed specifically for MET students, as traditional ML courses developed for computer science or engineering science programs do not align well with MET education.

2.2 Instructional Goals for an Application-Focused ML Course

The instructional goals for the machine learning course were developed directly in response to the challenges identified in Section 2.1. Rather than adopting a traditional, theory-driven ML curriculum designed for computer science or engineering science programs, the goals emphasize practical ML literacy aligned with the educational background and professional focus of MET students. The intent is not to train students to develop new machine learning algorithms, but to prepare them to understand, apply, and evaluate ML techniques within applied engineering contexts. The essential AI/ML competencies for MET students are summarized in Figure 1. Based on these considerations, the course is designed to achieve the following goals:

- Develop an understanding of the role of ML in engineering systems: Enable students to recognize where machine learning can be effectively applied in mechanical and manufacturing engineering contexts and how ML components interact with physical systems, sensors, and processes.

- Build the ability to work with engineering data and ML workflows: Help students gain experience in handling datasets, following structured ML workflows, and interpreting model outputs in relation to engineering performance and decision making.

- Support applied use of ML tools without excessive theoretical burden: Emphasize conceptual understanding and practical application rather than mathematical derivations or algorithm development, allowing students to use ML tools effectively within their technical domain.

- Develop foundational programming skills needed for ML applications: Introduce sufficient coding experience to enable students to understand, modify, and troubleshoot ML-related scripts, while treating programming as a supporting skill rather than the primary focus of the course.

- Promote awareness of ML limitations and responsible use: Encourage students to understand the assumptions, constraints, and appropriate use of ML models in engineering practice, supporting informed and responsible decision making.

Together, these goals define the scope of a machine learning course designed for MET students and guide the decisions made in structuring the course, selecting instructional approaches, and designing assessments, as described in the following sections.

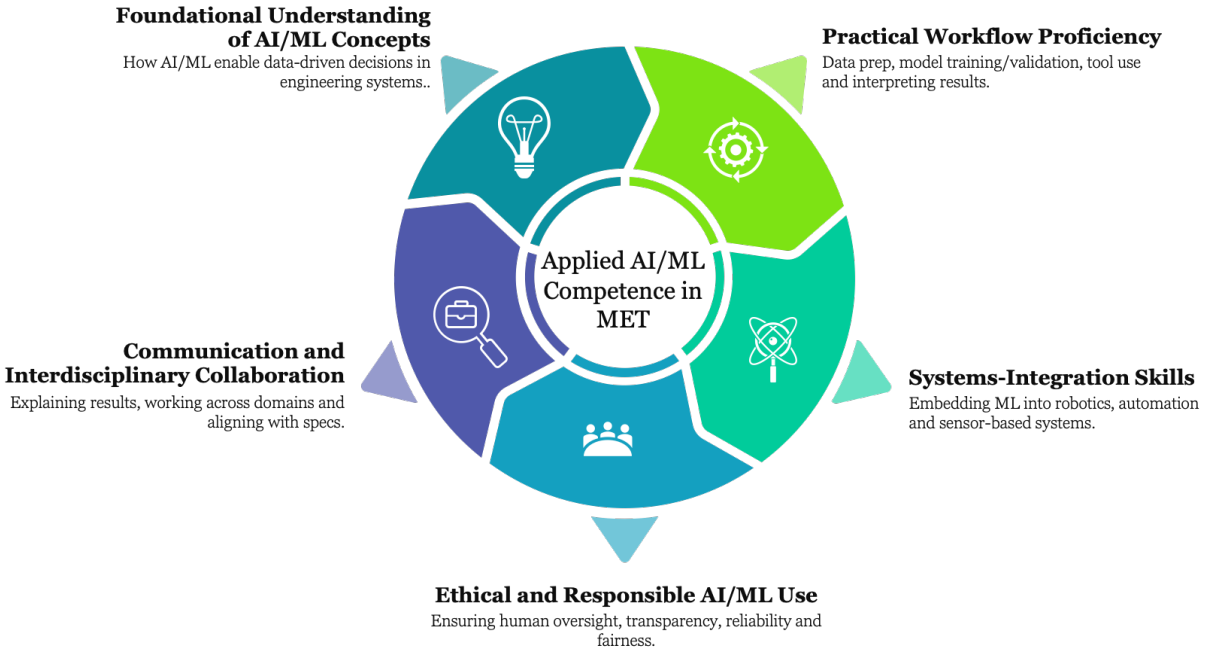

Figure 1. AI/ML competency framework for Mechanical Engineering Technology students

3. Design Philosophy

The design of the MET309 was guided by a set of instructional principles intended to align machine learning education with the background and learning needs of MET students. Rather than replicating traditional machine learning instruction from computer science or engineering science programs, the course was designed around the following principles:

- Alignment of lecture content with engineering technology practice: The selection of machine learning concepts covered in the course is guided by whether they are necessary for applying ML to real engineering problems commonly encountered in mechanical and manufacturing contexts. Lecture content focuses on introducing only those ML ideas that support practical problem solving and engineering decision making, helping students connect ML concepts to their engineering background and reinforcing their identity as applied engineers.

- Application-driven introduction of machine learning through hands-on laboratory activities: Machine learning concepts are introduced and reinforced primarily through structured laboratory exercises that emphasize hands-on experience. These lab activities are designed around engineering-relevant scenarios, allowing students to actively engage with data-driven methods and develop an intuitive understanding of how ML tools interact with physical systems and engineering processes.

- Emphasis on accessibility over theoretical depth: The course is designed with the assumption that students have completed only a statistics prerequisite and possess a very limited background in calculus. Mathematical derivations and detailed theoretical analysis of machine learning models are intentionally minimized. Instead, instruction emphasizes a visual and holistic understanding of how ML models function, how inputs influence outputs, and how model behavior relates to engineering applications, enabling students to engage with ML without being hindered by advanced mathematical requirements.

- Practice-oriented development of coding skills: The course is designed with the assumption that many students have little or no prior programming experience. Rather than introducing a dedicated Python programming module, coding skills are developed gradually through hands-on projects and laboratory activities. The emphasis is placed on understanding and modifying existing code and ML workflows, rather than writing code from scratch. Students are encouraged to leverage available machine learning libraries and tools to solve engineering problems, reflecting current industry practice and the growing role of AI-assisted coding tools in supporting implementation.

- Progressive development of ML-related skills: The course is designed as an introductory entry point to machine learning for MET students, with the recognition that it is not feasible to cover all major ML topics within a single course. Rather than emphasizing depth in individual algorithms, learning activities are organized to gradually increase in scope and complexity across the semester, focusing on building foundational ML knowledge and helping students develop confidence in working with ML tools and

workflows. This approach emphasizes that machine learning is accessible and manageable for engineering technology students, rather than an advanced or intimidating subject.

Together, these design principles guided decisions about how the course content was organized and sequenced. The following section describes the overall structure of the course and the major content areas included in the curriculum.

4. Course Structure and Content

MET309 is a 300-level course offered within the MET program. It is delivered over a 15-week semester with four contact hours per week, including three hours of lecture and one hour of laboratory instruction. The course has a single prerequisite in statistics and is designed for students with little or no prior experience in machine learning or computer programming.

The course integrates conceptual instruction with hands-on laboratory work, allowing students to gain practical exposure to ML workflows throughout the semester. Lecture sessions introduce key ideas and concepts, while laboratory sessions provide opportunities for students to work with data and apply ML tools in engineering-related contexts. Table 1 presents an overview of the week-by-week organization of the course, illustrating how lecture topics and laboratory activities are distributed across the semester. The following subsections describe the lecture content and laboratory components of the course in more detail.

4.1 Lecture Content and Conceptual Coverage

The lecture component of the course is organized into a sequence of thematic chapters that introduce core machine learning concepts commonly encountered in engineering applications. The major lecture topics are summarized below:

- Ch. 1 – Introduction to Machine Learning (Week 1): This chapter sets the stage by clarifying what AI and ML are (and how they differ from "rule-based" programming), then frames ML as a practical tool used across engineering workflows. It uses accessible, real-world examples (e.g., automation, predictive maintenance, and quality inspection) to help students quickly see "why ML matters" in mechanical and manufacturing contexts. The goal here is motivation and basic literacy, and students should walk away feeling that ML is relevant and approachable.
- Ch. 2 – Supervised Learning and Basic Classification (Weeks 2 & 3): This chapter introduces the idea of learning from labeled examples using plain, intuitive explanations. Classification is presented as a common engineering task (sorting, pass/fail decisions, pattern recognition), and the chapter uses a simple baseline classifier (e.g., perceptron-style logic) to illustrate how decision boundaries and "learning" work at a high level. The emphasis is on the workflow and interpretation, not math-heavy derivations.
- Ch. 3 – Regression Models (Week 4): This chapter focuses on prediction of continuous values using engineering-style examples (e.g., performance, energy use, wear trends, or other measurable variables). Students are introduced to linear regression and polynomial regression as practical tools and see how regression connects to real engineering decision

making. The chapter also reinforces the idea that a model is only useful if it can generalize to new cases, not just fit the training data.

- Ch. 4 – Data Preprocessing Pipeline (Weeks 5 & 6): This chapter covers the "before modeling" work that usually determines whether ML succeeds or fails in practice. It introduces common data issues and practical fixes—different data types, missing values, encoding categorical information, scaling, and splitting data for training/testing. The chapter frames preprocessing as an engineering task: getting sensor/process data into a form that tools can reliably use.
- Ch. 5 – Model Evaluation and Tuning (Weeks 7 & 8): This chapter explains why a single metric (especially accuracy) is often not enough, and how evaluation needs to match the engineering goal. Students learn the basic evaluation ideas used in practice for both classification and regression, and they discuss typical tradeoffs (for example, when different types of errors matter more). The chapter also introduces the idea of improving a model through tuning and repeated testing, in a way that feels like an engineering iteration loop.
- Ch. 6 – Unsupervised Learning (Week 9): This chapter introduces learning patterns from unlabeled data in an intuitive, application-driven way. Topics include clustering for grouping similar cases, dimensionality reduction for simplifying or visualizing data, and the general idea of discovering structure when "correct answers" are not available. The emphasis is on how these tools support real engineering work such as grouping defects, exploring sensor trends, or spotting unusual behavior.
- Ch. 7 – Deep Learning and Neural Networks (Weeks 10 & 11): This chapter introduces deep learning as an extension of machine learning for problems involving complex and high-dimensional engineering data. Students are introduced to the basic ideas behind artificial neural networks, including neurons, layers, and how layered structures enable models to learn increasingly abstract features. The focus is on understanding how neural networks differ from earlier ML models and when they are appropriate for engineering applications, rather than on mathematical formulation.
- Ch. 8 – Convolutional Neural Networks (CNNs) (Weeks 12 & 13): This chapter focuses on convolutional neural networks as a specialized class of deep learning models for structured, high-dimensional data. Students learn how core CNN components, such as convolutional layers, pooling operations, and hierarchical feature extraction, work together within a network. The chapter emphasizes why CNN architectures are effective for certain types of engineering data and how their structure differs from traditional neural networks.
- Ch. 9 – Recurrent Neural Networks (RNNs) (Weeks 14 & 15): The final chapter focuses on machine learning approaches for handling time-series and sequential data commonly encountered in engineering systems. Recurrent neural networks are introduced conceptually as models that account for temporal dependencies by allowing information from earlier steps to influence later predictions. Emphasis is placed on helping students understand how to organize, interpret, and model time-dependent data using RNN-based approaches, rather than on detailed algorithmic or mathematical treatment. Engineering-oriented examples illustrate how these models support analysis and prediction of dynamic system behavior.

Table 1. Course Outline for MET309: Machine Learning for Applied Engineering.

Date	Lecture	Lecture Learning Outcomes	Lab	Lab Learning Outcomes
Week 1	*Ch. 1 – Introduction to Machine Learning*	Identify the main types of machine learning (supervised, unsupervised, and reinforcement) and describe their key components and workflow in solving real-world problems.	*Lab 1: Python Crash Course*	Gain familiarity with Python syntax, data structures, and the Google Colab environment used for machine-learning development.
Week 2	*Ch. 2 – Supervised Learning and Basic Classification*	Explain how fundamental classification algorithms, such as the perceptron and decision trees, operate and compare their strengths and limitations in supervised learning tasks.	*Lab 2: Google Teachable Machine*	Use a no-code ML tool to understand the end-to-end classification workflow of training, testing, and evaluating a simple model.
Week 3			*Lab 3: Perceptron Classifier*	Implement and visualize a simple perceptron model to classify data points and observe how decision boundaries are formed.
Week 4	*Ch. 3 – Regression Models*	Describe how regression models, including linear, polynomial, and logistic regression models, are used to model and predict continuous or categorical outcomes.	*Lab 4: Comparing Classifiers*	Compare the performance of multiple supervised-learning algorithms and interpret model metrics to choose the most suitable classifier.
Week 5	*Ch. 4 – Data Preprocessing Pipeline*	Apply a complete data preprocessing workflow including data cleaning, encoding, feature scaling, selection, dimensionality reduction, and splitting to prepare datasets for modeling.	*Lab 5: Regression*	Apply linear and polynomial regression to predict continuous variables and analyze model accuracy using regression metrics.
Week 6			*Lab 6: ML Pipeline Development*	Construct a complete ML pipeline for data preprocessing, feature engineering, model training, validation, and testing.
Week 7	*Ch. 5 – Model Evaluation and Tuning*	Evaluate model performance using appropriate classification and regression metrics, recognize overfitting and underfitting, and apply cross-validation and hyperparameter tuning to improve model generalization.		
Week 8			*Lab 7: Regularization & Tuning*	Improve model generalization by applying regularization techniques and hyperparameter tuning using cross-validation.
Week 9	*Ch. 6 – Unsupervised Learning*	Explain how unsupervised learning methods, such as K-Means clustering, work and how they are used to identify patterns or groupings in data.	*Midterm Project – Traditional Machine Learning*	Independently apply traditional ML methods (classification, regression, or clustering) to an engineering dataset, demonstrating the ability to design, implement, and evaluate a full ML solution.
Week 10	*Ch. 7 – Deep Learning and Neural Networks*	Explain how artificial neural networks (ANNs) learn through layers, activation functions, and backpropagation, and apply them to model nonlinear relationships in data.	*Lab 8: Clustering*	Use unsupervised-learning methods such as K-Means to group unlabeled data and interpret clustering results.
Week 11			*Lab 9: Neural Network for Classification*	Build a basic artificial neural network in TensorFlow or PyTorch to classify data and observe how model parameters affect performance.
Week 12	*Ch. 8 – Convolutional Neural Networks (CNNs)*	Describe how convolution, pooling, and feature extraction enable CNNs to perform image recognition and object detection tasks.		
Week 13			*Lab 10: CNN for Image-Based Classification*	Implement a convolutional-neural-network model to perform image classification tasks and visualize feature extraction in layers.
Week 14	*Ch. 9 – Recurrent Neural Networks (RNNs)*	Explain how RNNs capture temporal dependencies in sequential data using hidden states and gated architectures such as LSTM and GRU.	*Lab 11: RNN for Time-Series Prediction*	Develop a recurrent-neural-network model to analyze sequential data and predict time-dependent trends.
Week 15			*Final Project – Deep Learning*	Design and complete an independent deep-learning project that applies neural-network techniques to an engineering problem, demonstrating integration of conceptual knowledge and hands-on skills.

4.2 Laboratory Projects and Hands-On Activities

The laboratory component of the course provides hands-on experience that directly supports the lecture topics through a sequence of structured lab projects. All labs are conducted using Google Colab [10], allowing students to work in a consistent, cloud-based Python environment without the need for local software installation. This setup reduces technical overhead and enables students to focus on applying machine learning concepts rather than managing computing resources.

Each laboratory project is aligned with the corresponding lecture chapter and is designed around realistic engineering problems and datasets. The labs guide students through complete machine learning workflows, including data preparation, model application, and result interpretation, reinforcing how lecture concepts are used in practical engineering contexts. Rather than serving as isolated coding exercises, the labs are intended to connect machine learning methods to familiar engineering tasks and decision-making processes.

The laboratory sequence is intentionally organized to gradually increase the level of coding responsibility. Early labs provide substantial structure through guided notebooks and predefined code, while later labs require students to take a more active role in modifying code and configuring models. This progression allows students with limited programming background to build confidence over time while maintaining an emphasis on understanding and applying machine learning tools rather than developing code from scratch. Alongside the regular lab exercises, the course includes two term projects that ask students to work more independently. These projects move beyond guided notebooks and require students to apply machine learning methods to less structured engineering problems. A summary of the individual laboratory projects is provided below:

- Lab 1 – Python Crash Course (Week 1): This lab introduces students to the Python programming environment used throughout the course, with emphasis on basic syntax, data structures, and common libraries relevant to machine learning workflows. Rather than teaching Python as a standalone topic, the lab focuses on helping students read, understand, and modify existing code in a Google Colab notebook. The goal is to establish baseline familiarity with Python needed for subsequent ML labs.

- Lab 2 – Google Teachable Machine (Week 2): This lab introduces machine learning concepts through a no-code interface using Google Teachable Machine [11]. Students create and test simple classification models using image or sensor-like data, allowing them to experience the full ML workflow without writing code. The lab emphasizes intuition, experimentation, and interpretation of results before transitioning to Python-based implementation.

- Lab 3 – Perceptron Classifier (Week 3): In this lab, students implement a simple perceptron-based classifier using a well-known dataset. The lab introduces supervised learning in a Python environment while maintaining a high level of guidance through structured notebooks. Students focus on understanding relationships between input features, model parameters, and predictions.

- Lab 4 – Comparing Classifiers (Week 4): This lab expands on classification by comparing multiple machine learning models using a common dataset. Students apply different classifiers available in scikit-learn and evaluate their performance using appropriate metrics. The lab emphasizes model comparison, result interpretation, and the idea that different models may be suitable for different engineering problems.

- Lab 5 – Regression (Week 5): This lab introduces regression analysis using an engineering-relevant dataset with continuous-valued outputs. Students apply linear and polynomial regression models to explore relationships between variables and assess prediction performance, reinforcing regression concepts from lecture.

- Lab 6 – ML Pipeline Development (Weeks 6 & 7): This lab guides students through the development of a complete machine learning pipeline using a realistic engineering dataset. The lab emphasizes the full workflow, including data inspection, data quality handling, outlier detection, data splitting, feature encoding, scaling, feature selection, dimensionality reduction, and model evaluation. Students work through these steps in a structured sequence that mirrors real-world engineering practice and highlights the importance of data preparation and workflow design.

- Lab 7 – Regularization & Model Generalization (Week 8): This lab focuses on improving model generalization through regularization techniques applied to regression problems. Students explore how model complexity affects performance and learn how regularization helps balance accuracy and robustness when working with engineering data.

- Midterm Project – Classical Machine Learning (Week 9): The midterm project requires students to independently apply classical machine learning methods to a real engineering dataset. Students choose between a classification or regression problem, prepare the data, select appropriate models, evaluate performance, and interpret results. The project emphasizes independent problem formulation and reinforces core ML workflows.

- Lab 8 – Clustering (Week 10): This lab introduces unsupervised learning through clustering analysis using an engineering-relevant dataset. Students apply clustering techniques to explore patterns and groupings in unlabeled data, emphasizing interpretation and exploratory analysis in engineering contexts.

- Lab 9 – Neural Networks for Engineering Prediction (Weeks 11 & 12): This lab introduces neural networks through an engineering-focused prediction task using a real-world dataset. Students apply a multi-layer perceptron model to learn complex relationships between inputs and outputs, focusing on training behavior, performance, and prediction accuracy.

- Lab 10 – CNN for Image Classification (Week 13): This lab revisits image classification using convolutional neural networks. Students transition from no-code approaches to CNN-based models implemented in Python, with emphasis on training, evaluation, and the impact of hyperparameter choices on performance.

- Lab 11 – RNN for Time Series Prediction (Week 14): This lab introduces recurrent neural networks through an engineering-focused time-series prediction task. Students organize data into sequential inputs and apply RNN models to capture temporal relationships, emphasizing when sequential models are appropriate for dynamic engineering systems.

- Final Project – Deep Learning (Week 15): The final project extends the midterm project by revisiting the same engineering problem using deep learning methods. Students remain on the same classification or regression track and replace their classical model with a neural network–based approach. The project emphasizes comparison between traditional and deep learning models and highlights practical tradeoffs in model selection.

Together, these laboratory projects and term projects provide students with repeated exposure to machine learning workflows applied to real engineering problems. Throughout the course, datasets are drawn from publicly available engineering-focused repositories, including the UCI Machine Learning Repository [12] and Kaggle [13], which offer a wide range of realistic datasets suitable for machine learning education. Using these resources reinforces that meaningful, application-focused ML learning can be achieved using accessible, real-world engineering data.

5. Instructional Methods for MET Students

When teaching machine learning to MET students, several challenges become immediately apparent. Many students have limited preparation in advanced mathematics, little prior experience with programming, and can feel overwhelmed when introduced to a large number of unfamiliar machine learning concepts within a single course. The instructional methods used in this course are intentionally designed to address these challenges and to help students view machine learning as an accessible and practical tool for engineering applications rather than an abstract or intimidating subject.

- Addressing limited mathematical background through concept-driven instruction: Machine learning concepts are introduced using intuitive explanations, visualizations, and engineering-oriented examples rather than mathematical derivations. For example, gradient descent is presented visually as an iterative process of reducing prediction error, using plots of loss versus training iterations rather than calculus-based optimization. Similarly, backpropagation is introduced conceptually by illustrating how prediction errors are passed backward through network layers to update model parameters, without deriving update equations. In addition, instruction emphasizes helping students understand which types of models and parameters are appropriate for different engineering problems, such as when a simple regression model is sufficient versus when a neural network is more appropriate, or how choices like learning rate, model complexity, and regularization affect performance. The course assumes statistics as the primary prerequisite, with only minimal reliance on calculus, and focuses on interpreting model behavior and results in an engineering context rather than mathematical theory.

- Supporting limited coding experience through a staged and assisted approach: The course begins with a short Python crash course to introduce basic syntax, data structures, and the

coding environment needed for early laboratory activities. Initial lab projects emphasize observing and explaining the behavior of given code, with students focusing on interpreting outputs and understanding how changes in inputs affect results. As the semester progresses, labs shift to modifying parameters and configuration settings within predefined code, followed by exercises where students complete major workflow steps using guided comments and structured prompts. This gradual transition allows students to move from observation to partial implementation without being overwhelmed. Around Week 5, a more focused Python review is incorporated alongside the introduction of major machine learning libraries, allowing students to revisit and strengthen their coding skills in a context where they can immediately see their relevance. The sequence culminates in independent midterm and final projects, where students apply machine learning methods with greater autonomy. Throughout the course, students are explicitly permitted to use generative AI tools such as ChatGPT (OpenAI) [14] to assist with code interpretation and debugging, reinforcing coding as a support skill rather than a barrier to learning machine learning.

- Managing overwhelming ML content through selective coverage and continuity: Given the constraints of a single-course format, the course focuses on a small set of foundational machine learning models that are sufficient for solving many practical engineering problems, rather than attempting broad algorithmic coverage. Concepts are consistently tied to real-world engineering examples to reduce abstraction; for instance, different model evaluation metrics are discussed using concrete scenarios where certain types of errors matter more than others. Key ideas such as overfitting, underfitting, and gradient-based learning are revisited repeatedly across multiple chapters and labs, allowing students to reinforce understanding through repeated exposure rather than isolated instruction. Classroom discussions are actively encouraged during and after lectures, allowing students to compare interpretations, ask questions, and reflect on how ML concepts apply to engineering situations. To further manage cognitive load, each chapter concludes with a short concept-focused quiz, typically in a multiple-choice format, to help students reflect on and consolidate essential ideas before moving on. Together, these strategies help students engage with machine learning concepts without feeling overwhelmed while building a coherent understanding across the semester.

Together, these instructional methods create a learning environment intentionally aligned with the background, learning preferences, and professional goals of MET students. The effectiveness of these methods is evaluated through an assessment plan, which is described in the following section.

6. Assessment Design

The assessment design for MET309 evaluates students' conceptual understanding, applied skills, and ability to use machine learning tools in engineering contexts. Instead of relying on traditional exam-based assessment, the course uses a combination of quizzes, laboratory assignments, term projects, and a post-course survey to capture student learning and engagement across multiple dimensions.

Student performance in the course is evaluated using the following components:

- Chapter Quizzes (40%): Short, concept-focused quizzes are administered after each major lecture chapter. These quizzes primarily use multiple-choice questions to assess students' understanding of key ideas such as model behavior, overfitting and underfitting, evaluation metrics, and appropriate model selection. The quizzes reinforce essential concepts without emphasizing mathematical derivation or programming syntax.

- Laboratory Assignments (40%): A sequence of structured laboratory projects forms the core of the course assessment. Each lab evaluates students' ability to apply machine learning workflows to engineering datasets, interpret results, and explain model behavior. Grading emphasizes correct use of methods, reasoning, and interpretation rather than code complexity, allowing students to demonstrate steady development over the semester.

- Midterm and Final Projects (20%): Students complete an independent, two-stage term project centered on a real engineering problem, selecting either a classification or regression task. The midterm stage focuses on applying classical machine learning methods, while the final project revisits the same problem using deep learning approaches. The project evaluates students' ability to integrate concepts learned across multiple chapters and apply them within a complete machine learning workflow, including data preparation, model training, performance evaluation, and basic deployment-oriented considerations. Assessment emphasizes integration of knowledge, interpretation of model behavior, and justification of design choices across the full project lifecycle.

No midterm or final exams are administered in this course. This reflects the applied nature of machine learning practice, where learning is demonstrated through iterative experimentation, data analysis, and interpretation rather than time-limited problem solving.

At the end of the semester, a student survey was administered to gather feedback on learning experiences, confidence in applying machine learning concepts, and perceptions of course components. While the survey does not contribute to the course grade, it provides additional perspective to support course evaluation and future refinement.

Overall, this assessment approach emphasizes continuous, application-based learning and aligns with the goal of preparing MET students to apply machine learning as a practical engineering tool.

7. Results and Observations

MET309 was first offered in Fall 2025 as a new elective course in the MET program. The course enrolled 13 students, and all results reported in this section are based on data collected from this initial offering. Because the course is newly introduced and enrollment is still limited, the results should be interpreted as preliminary. The small sample size reflects both the elective nature of the course and the early stage of implementation, rather than a lack of student interest. While the dataset does not support broad generalization, the assessment results, project outcomes, and student feedback provide useful insight into how MET students respond to an application-focused approach to machine learning instruction.

7.1 Quiz Performance

Student understanding of lecture concepts was measured using nine in-class quizzes, corresponding to the nine major lecture chapters. Each quiz was closed-book and administered during the class meeting immediately following completion of the chapter, with the exception of the final chapter, whose quiz was given on the last day of the semester.

Each quiz contained at least 15 multiple-choice questions designed to assess students' understanding of the most important machine learning concepts introduced in the lecture. The questions were intentionally written to be conceptually challenging, focusing on interpretation and reasoning rather than recall. As a result, quiz averages are influenced by question difficulty and should be interpreted as a measure of students' grasp of lecture material rather than memorization of facts.

Table 2 summarizes the class average scores for the nine quizzes. Average scores range from 71.0% to 81.3%, with most quizzes falling in the mid-70% to low-80% range. While individual quiz averages vary by topic, performance remains relatively consistent across chapters.

Table 2 Average Scores for Quizzes

Quizzes	*#1*	*#2*	*#3*	*#4*	*#5*	*#6*	*#7*	*#8*	*#9*
Average Score	75.0%	79.2%	76.1%	73.7%	71.0%	74.8%	80.2%	81.3%	77.8%

Taken together, these results indicate that students were generally able to follow and understand the machine learning concepts presented in lecture throughout the semester. The absence of large fluctuations in average scores suggests that the pacing and depth of lecture content were appropriate for MET students, even as new and unfamiliar topics were introduced.

7.2 Laboratory Performance

Student performance on laboratory assignments was evaluated through a sequence of eleven structured lab projects, each aligned with a corresponding lecture topic. Unlike quizzes, which focused on conceptual understanding, the laboratory component emphasized applied skills, interpretation of results, and completion of machine learning workflows using realistic engineering datasets.

The laboratory assignments were intentionally designed with varying levels of guidance. Early labs provided predefined code and structured notebooks that asked students to observe results and explain model behavior. As the semester progressed, labs gradually required students to make parameter changes, modify existing code, and complete missing workflow steps using guided comments. This structure was intended to reduce barriers related to programming while allowing students to focus on understanding how machine learning methods operate in practice. Students were permitted to use large language model (LLM) tools, such as ChatGPT, to assist with code interpretation and debugging, consistent with the course emphasis on applied understanding rather than code development from scratch.

For each lab assignment, students were required to submit both a completed Google Colab notebook and a written report. Detailed grading rubrics were provided for each lab, with evaluation focusing on correctness of workflow execution, quality of interpretation, and clarity of explanation

rather than code originality. This dual-submission requirement helped distinguish between mechanical completion of tasks and students' understanding of the underlying machine learning concepts.

Table 3 summarizes the class average scores for the laboratory assignments. Average scores range from 84.2% to 100.0%, with most labs falling in the high-80% to low-90% range. While some variation in average scores is observed, these differences largely reflect variations in lab difficulty and scope rather than changes in student engagement or effort. Later labs, which involved more complex workflows and reduced scaffolding, did not show a systematic decline in performance.

Table 3 Average Scores for Labs

Labs	*#1*	*#2*	*#3*	*#4*	*#5*	*#6*	*#7*	*#8*	*#9*	*#10*	*#11*
Average Score	100.0%	92.3%	84.2%	87.7%	89.3%	88.1%	92.2%	86.9%	91.6%	93.4%	90.5%

Overall, laboratory results indicate that students were able to successfully complete application-focused machine learning tasks within the structured environment provided by the course. Given the guided nature of many lab activities and the use of support tools, lab performance reflects students' ability to follow, apply, and interpret machine learning workflows within a structured environment, rather than fully independent problem solving.

7.3 Term Project Outcomes

The midterm and final projects were designed as independent, end-to-end machine learning projects intended to assess students' ability to combine concepts learned across the course and apply them to real-world engineering problems. Students selected either a classification or regression task and were provided with a set of candidate engineering datasets and a clearly defined list of required tasks. These tasks covered the full machine learning workflow, including data preparation, model training, evaluation, and interpretation.

Project assessment was based on three components: completion of required tasks, the submitted Google Colab notebook containing the implemented workflow, and a written report explaining model choices, results, and observations. Detailed rubrics were provided to guide expectations and ensure consistency in evaluation. While students were permitted to use LLM tools to assist with coding and debugging, grading emphasized task completion, correctness of workflow, and clarity of explanation rather than code originality.

The average score for the midterm project was 88.7%, and the average score for the final project was 90.1%. These averages are not intended to be compared directly, as the midterm and final projects used different evaluation rubrics reflecting differences in project scope and technical focus. Instead, the scores indicate that students were able to meet the expectations of each project stage. In particular, most students successfully completed the required tasks for both classical and deep learning–based workflows, maintaining a coherent project structure and providing meaningful interpretation of results when revisiting the same engineering problem.

7.4 Student Feedback and Reflections

At the end of the semester, a student survey was administered to collect feedback on learning experiences, confidence, and overall perceptions of the course. The survey began with two questions asking students to self-report their background prior to enrolling in MET309:

- Before this course, how would you rate your experience with Python programming?
- Before this course, how familiar were you with machine learning or AI?

The results from these background questions are summarized in Figure 2. Student responses indicate that most participants entered the course with limited prior programming experience. Similarly, most students reported limited exposure to machine learning before enrolling in the course. This background distribution is consistent with the typical preparation levels observed among MET students.

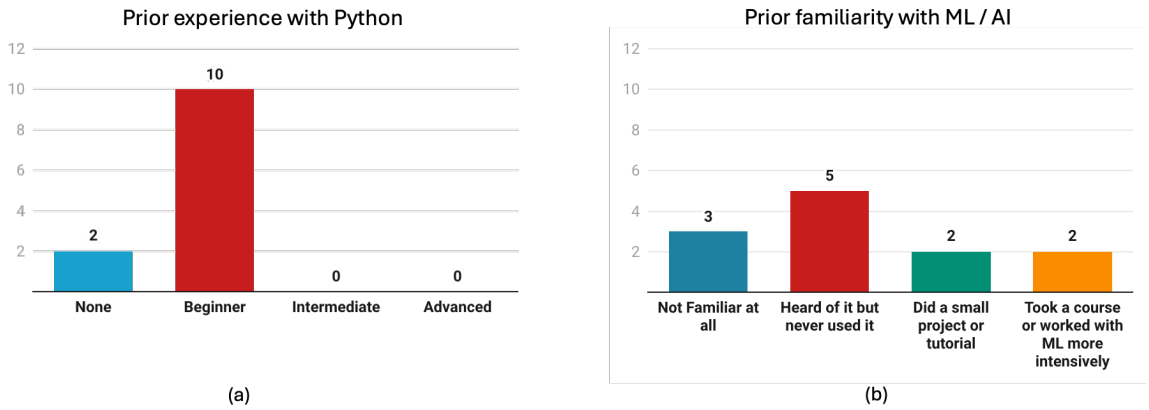

Figure 2. Self-Reported Student Background in (a) Python Programming and (b) Machine Learning Prior to MET309

Following the background questions, the survey asked students to rate their perceived understanding of machine learning topics covered in the lecture portion of the course. These questions were aligned with the major lecture chapters and emphasized conceptual understanding rather than mathematical derivation or coding proficiency. Students rated their understanding using a five-point Likert scale ranging from Poor (1) to Excellent (5). The results are summarized in Table 4.

Overall, the self-evaluation results indicate a clear pattern in perceived understanding across topics of varying complexity. Students reported higher confidence in foundational machine learning concepts and core workflows, including problem formulation, model selection, data preparation, and interpretation of results. These topics correspond to the early and middle portions of the course, where concepts are reinforced through repeated exposure in lectures, quizzes, and laboratory activities. Lower average scores were reported for more advanced topics, particularly those related to deep learning architectures and internal model mechanisms. This trend is expected given the introductory scope of the course and the limited mathematical background typical of MET students.

Table 4 Self-Evaluation Scores for the Perceived Understanding of Lecture Topics

	Self-Evaluation Questions	Average Scores (Excellent = 5; Good = 4, Average = 3, Below Average = 2, Poor = 1)
1	Understanding the three main ML types (Supervised, Unsupervised, Deep Learning)	4.00
2	Identifying practical engineering applications of ML	3.82
3	Distinguishing between classification and regression tasks	4.18
4	Understanding linear and polynomial regression models	3.82
5	Understanding regression evaluation metrics (MSE, MAE, RMSE, R^2)	3.27
6	Understanding the ML workflow (train/test split, fit → evaluate → predict)	3.72
7	Understanding data preprocessing (scaling, encoding, missing values)	3.72
8	Understanding hyperparameter tuning (grid search, random search)	3.09
9	Understanding confusion matrix and ROC-AUC	3.36
10	Understanding overfitting, underfitting, and generalization	3.81
11	Understanding cross-validation	3.72
12	Understanding K-Means clustering	3.00
13	Understanding Hierarchical Clustering and linkage/dendrograms	3.18
14	Understanding DBSCAN and density-based clustering	3.00
15	Interpreting silhouette scores and Davies–Bouldin Index (DBI)	2.82
16	Understanding artificial neurons and activation functions (ReLU, sigmoid, softmax)	2.72
17	Understanding neural network loss functions (MSE, cross-entropy)	2.55
18	Understanding feedforward and backpropagation (conceptual)	2.72
19	Understanding Multi-Layer Perceptrons (MLPs)	2.64
20	Interpreting neural network training curves (loss, accuracy)	3.00
21	Understanding the role of convolutional and pooling layers in CNN-based image classification	2.64
22	Understanding how RNNs capture sequential or time-dependent patterns using hidden states	3.00
23	Overall difficulty of ML lecture topics this semester	3.18

In addition to lecture-related topics, the survey asked students to evaluate their learning experiences and performance related to the laboratory component of the course. These questions focused on students' confidence in completing lab tasks, using computational tools, and applying machine learning methods in hands-on settings. The results are summarized in Table 5.

Overall, students reported good confidence in completing laboratory projects. Self-evaluation scores indicate that most students were comfortable following lab workflows, preparing datasets, and training models, while slightly lower confidence was reported for interpreting results and applying methods more independently in later labs.

Table 5 Self-Evaluation Scores for the Performance in Completing Lab Projects

	Self-Evaluation Questions	Average Scores (Excellent = 5; Good = 4, Average = 3, Below Average = 2, Poor = 1)
1	My ability to complete lab projects successfully is:	4.18
2	My confidence in using Google Colab and Jupyter Notebook is:	4.25
3	My ability to learn and use Python in lab sessions is:	3.83
4	My ability to learn and use Python packages (general ML-related libraries) is:	3.75
5	My ability to prepare and preprocess datasets in labs is:	3.92
6	My ability to train, evaluate, and modify ML models in labs is:	3.75
7	My ability to interpret model outputs and evaluation metrics in labs is:	3.58
8	My ability to follow lab workflows and connect them to lecture topics is:	3.83
9	My ability to apply ML techniques to simplified engineering problems in labs is:	3.58
10	Overall self-evaluation of my performance in lab projects:	3.80

Student feedback also addressed perceptions of the overall course structure, including the organization of lecture topics and the alignment between lectures and laboratory activities. As shown in Table 6, students generally rated the lecture component positively in terms of relevance to engineering applications, topic sequencing, and overall workload. Average scores indicate that students felt the depth of coverage was appropriate for an introductory machine learning course and that lecture materials supported their understanding. Students also reported increased awareness of how machine learning is used in engineering practice and greater confidence in thinking about engineering problems using data-driven approaches.

Feedback related to the laboratory component was consistently strong. Table 6 shows high average scores for lab–lecture alignment, clarity of lab instructions, and the extent to which labs reinforced conceptual understanding through practical application. Students also indicated that the lab activities provided meaningful engineering-relevant experience and helped build confidence in applying machine learning methods. Slightly lower ratings related to the balance between lecture and lab time and independent application suggest opportunities for continued refinement, particularly in supporting the transition from guided exercises to more open-ended work.

Overall, the course structure feedback summarized in Table 6 suggests that the integrated lecture–lab design effectively supports applied ML learning for MET students while providing useful guidance for future course improvements.

Given that limited mathematical preparation and programming experience are widely recognized challenges for MET students, the survey included specific questions to evaluate whether the course achieved an appropriate balance in these areas. The results of these questions are summarized in Figure 3.

As shown in Figure 3(a), most students indicated that the amount of mathematics in the course was appropriate, with only a small number reporting that it was slightly too much. Similarly, Figure 3(b) shows that the coding workload was viewed as appropriate by the majority of students, with minimal responses indicating concern about excessive or insufficient coding. These responses

suggest that the course design successfully addressed two of the primary barriers often associated with introducing machine learning in MET curricula.

Feedback on instructional support for learning coding, shown in Figure 3(c), indicates that students generally viewed the level of support positively. Most responses fall between "Adequate" and "Excellent," suggesting that the use of templates, explanations, and guided examples helped students engage with coding tasks without allowing programming difficulty to overshadow machine learning concepts.

Table 6 Feedback Scores for Course Structure Design

	Self-Evaluation Questions	Average Scores (Excellent = 5; Good = 4, Average = 3, Below Average = 2, Poor = 1)
	Lecture Design	
1	The topics covered in this course are useful for engineering applications.	3.83
2	The order of topics created a logical and effective learning progression.	4.08
3	The depth of each topic was appropriate for an introductory ML course in engineering.	4.0
4	The lecture materials supported my understanding effectively.	3.92
5	The lecture content connected well to real engineering problems.	3.75
6	The overall content workload was reasonable.	4.36
7	The course increased my awareness of how ML is used in engineering industries.	3.91
8	The course improved my ability to think about engineering problems using data-driven methods.	4.17
	Lab Design	
1	The labs were well aligned with the lecture topics.	4.33
2	The labs reinforced and deepened my understanding of ML concepts.	4.58
3	The labs provided practical experience relevant to engineering applications.	4.33
4	The lab instructions were clear and easy to follow.	4.5
5	The labs increased my confidence in applying ML independently.	4.08
6	The balance between lecture time and lab time was appropriate.	3.67

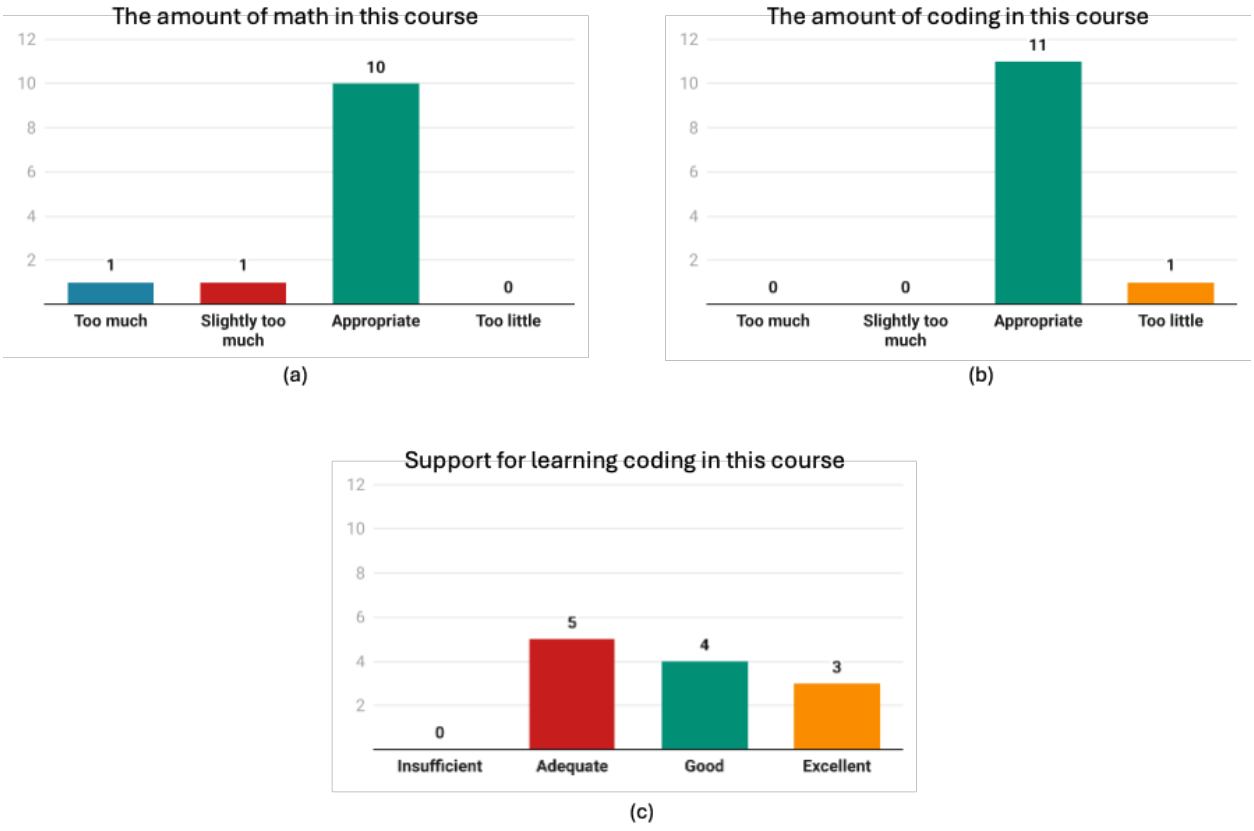

Figure 3. Student Feedback on the (a) Balance of Mathematical Content, (b) Coding Workload, and (c) Instructional Support in MET309

The survey also asked students to identify challenges they encountered during the course. Student responses to this question are summarized in Figure 4. The most frequently reported challenge was keeping up with theoretical concepts, followed by lack of prior experience in coding. These results are consistent with the background profile of MET students and reflect the introduction of unfamiliar machine learning concepts within a single semester. In contrast, relatively few students identified mathematical foundations as a primary challenge, suggesting that the emphasis on conceptual explanations rather than mathematical derivations helped reduce this barrier. Difficulty connecting lecture content to laboratory activities and managing weekly workload were reported less frequently, indicating that the integrated lecture–lab structure and pacing were generally effective. These results reinforce the importance of staged coding support and guided laboratory design in addressing persistent coding-related challenges for MET students.

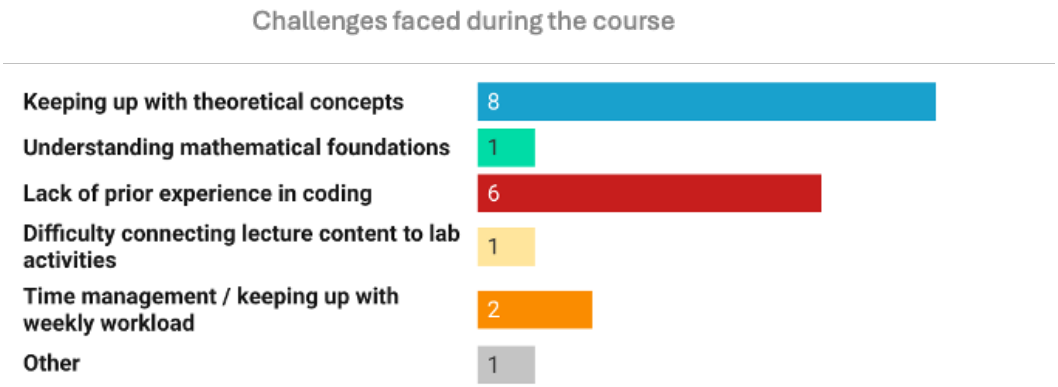

Figure 4. Student-Reported Challenges Encountered During MET309

Students were also asked to compare the overall difficulty of the course to their expectations at the beginning of the semester. As shown in Figure 5, most students reported that the course difficulty was about what they expected or slightly easier. Only a small number of students found the course more difficult than expected, and no students reported it as much more difficult. This result suggests that, despite the challenges associated with coding and unfamiliar machine learning concepts, the course structure and instructional support helped students manage expectations and remain engaged throughout the semester.

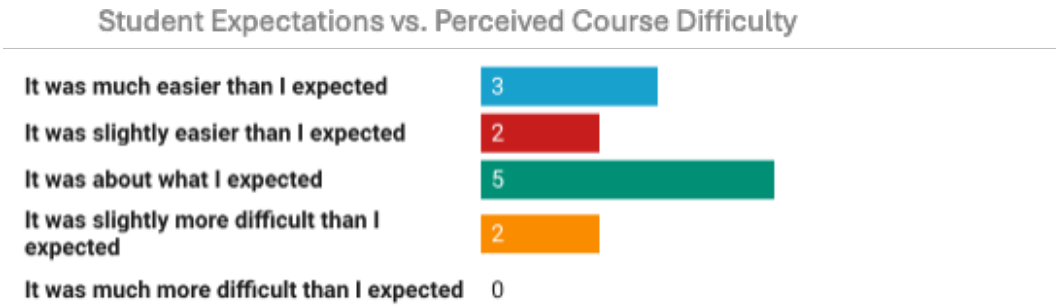

Figure 5. Student Perceptions of Overall Course Difficulty Compared to Initial Expectations

The final survey questions examined students' perceived preparedness to apply machine learning after completing MET309 and their likelihood of using ML or data-driven methods in future academic or professional work. As shown in Figure 6, most students reported feeling prepared or somewhat prepared to use machine learning after the course, with very few indicating low preparedness. In addition, a majority of students indicated that they are likely or very likely to use machine learning in future senior projects, employment, or graduate studies. These results suggest that, despite limited prior background, students completed the course with increased confidence and willingness to apply machine learning as a practical engineering tool.

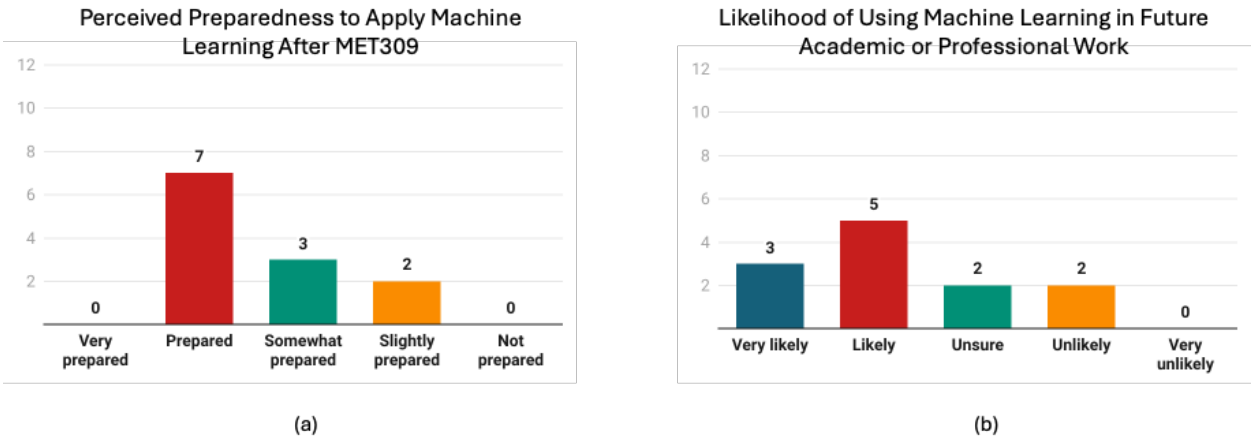

Figure 6. (a) Student self-reported preparedness to apply machine learning after completing MET309 and (b) likelihood of using machine learning or data-driven methods in future academic or professional work.

While these findings are based on a small cohort, the consistency across assessments and survey responses provides encouraging initial evidence for the effectiveness of the course design.

8. Discussion and Conclusion

The assessment results from quizzes, laboratory assignments, term projects, and student surveys indicate that MET students were able to engage with machine learning concepts and workflows within a single, application-focused course. Across the semester, students demonstrated consistent understanding of lecture material, strong performance in structured laboratory activities, and the ability to complete end-to-end machine learning projects using real engineering datasets. Student feedback further confirms that the course design helped make machine learning accessible despite limited prior exposure to mathematics, programming, and AI-related topics.

From an instructional perspective, several challenges emerged during the delivery of the course. Differences in student preparation, particularly in programming experience, remained the most persistent issue. Although staged coding support, guided notebooks, and the use of LLM tools reduced entry barriers, coding continued to require substantial effort for some students. In addition, even with careful topic selection and an emphasis on foundational, engineering-relevant models, the number of unfamiliar machine learning concepts introduced within a single semester was still perceived as overwhelming by some students, especially in later deep learning topics.

Despite these challenges, the course demonstrates that a focused, application-driven machine learning course can be successfully developed for MET students. MET309 achieved its primary goal of introducing machine learning as a practical engineering tool rather than an abstract theoretical subject. The instructional design directly addressed two major barriers commonly faced by MET students, limited mathematical background and limited programming experience, through concept-driven lectures, staged coding progression, and guided laboratory workflows. Assessment outcomes and student feedback suggest that these strategies were effective in supporting learning while maintaining appropriate rigor.

At the same time, the experience highlights important directions for future improvement. Further refinement of content pacing and scope is needed to better manage cognitive load within a single course. In addition, providing a programming-focused course prior to or alongside MET309 would likely enhance student readiness and allow deeper engagement with machine learning methods. Finally, the strong motivation and effort demonstrated by the students played a critical role in the success of the course and reflect a growing interest among MET students in acquiring machine learning skills for modern engineering practice.

Taken together, this initial offering of MET309 illustrates a practical approach to introducing machine learning within engineering technology programs. Ongoing refinement of content scope, pacing, and prerequisite preparation will further strengthen the course, while the current design provides a concrete starting point for similar programs seeking to integrate applied machine learning into engineering technology curricula.

References

1 Wuest, T., Weimer, D., Irgens, C., & Thoben, K.-D. (2016). Machine learning in manufacturing: Advantages, challenges, and applications. Production & Manufacturing Research, 4(1), 23–45.

2 Lee, J., Davari, H., Singh, J., & Pandhare, V. (2018). Industrial artificial intelligence for industry 4.0-based manufacturing systems. Manufacturing Letters, 18, 20–23.

3 Tao, F., Qi, Q., Liu, A., & Kusiak, A. (2018). Data-driven smart manufacturing. Journal of Manufacturing Systems, 48, 157–169.

4 Carvalho, T. P., Soares, F. A. A. M. N., Vita, R., Francisco, R. da P., Basto, J. P., & Alcalá, S. G. S. (2019). A systematic literature review of machine learning methods applied to predictive maintenance. Computers & Industrial Engineering, 137, 106024.

5 Manufacturing Leadership Council. (2023). The Future of Industrial AI in Manufacturing [PDF]. Available: https://www.manufacturingleadershipcouncil.com/wp-content/uploads/2023/06/The-Future-Of-AI-In-Manufacturing-MLC-2023.pdf

6 The Institution of Engineering and Technology (IET). (2023). Skills for a Digital Future 2023 Survey [PDF]. Available: https://www.theiet.org/media/11064/skills-for-a-digital-future-survey.pdf

7 Li, W., Hung, Y., Guttman, R., Zhang, S. and Yu, N., (2024). Designing an AI-Enhanced Module for Robotics Education in Mechanical Engineering Technology. In 2024 Fall ASEE Mid-Atlantic Section Conference, Farmingdale, NY.

8 Li, W., Hung, Y., Altuger-Genc, G., Zhang, S., Tatoglu, A. and Zhang, Z., (2025). Integrating AI/ML Learning in Senior Projects for Mechanical Engineering Technology Students. In 2025 ASEE Annual Conference & Exposition, Montreal, Canada.

9 Li, W., Hung, Y., Zhang, Z., Yu, N., Zhang, J. and Ladeinde, F., (2025). Incorporating Machine Learning into the MET Curriculum. In 2025 Fall ASEE Mid-Atlantic Section Conference, Stony Brook, NY.

10 Google, “Google Colaboratory,” Google Research, 2023. Available: https://colab.research.google.com

11 Google, “Teachable Machine,” Google Creative Lab, 2023. Available: https://teachablemachine.withgoogle.com

12 D. Dua and C. Graff, “UCI machine learning repository,” University of California, Irvine, School of Information and Computer Sciences, 2019. Available: https://archive.ics.uci.edu/ml

13 Kaggle, “Kaggle: Your home for data science,” Kaggle Inc., 2023. Available: https://www.kaggle.com

14 OpenAI, “ChatGPT,” OpenAI, 2023. Available: https://chat.openai.com