Tuning the Parameters: A Maritime-Tuned Machine Learning Course

Mr. Vincenzo Antonio Ventricelli, SUNY Maritime College

Vincenzo Ventricelli is an undergraduate student and student researcher at the State University of New York Maritime College pursuing a bachelor's degree in Electrical Engineering and a USCG Unlimited License. The focus of his current research is the applications of machine learning in the maritime industry, including the use of maritime-related datasets in the classroom. In addition to machine learning, he has a deep interest in other electrical engineering-related topics such as communications theory, control engineering, and power distribution.

Dr. Paul M. Kump, SUNY Maritime College

Dr. Paul M. Kump joined SUNY Maritime College in 2012 and is Associate Professor of Electrical Engineering. His research interests are in the areas of machine learning (ML), signal processing, and alternative teaching strategies. Dr. Kump has developed intelligent software-defined radio for the US Navy in electronic warfare, nuclear material detection algorithms for the US government at US seaports, and crime prediction software for the Chicago Police Department. He recently collaborated with Mount Sinai Hospital to create smart software for automatic error detection in patient radiation therapy treatment plans. In his spare time, Dr. Kump works to combine his research with his love of electronic music performance, teaching machines the craft of songwriting. With extensive course and curriculum design experience, Dr. Kump is continuously committed to developing engineering programs that best prepare students for the ever-changing demands of industry leaders. His teaching interests include online and HyFlex education, as well as classroom flipping and education research-based tasks. He created Maritime College's ENGR 396 Machine Learning course and has been recognized by Open SUNY for excellence in online teaching, pioneering the School of Engineering's online course offerings.

Van-Hai Bui

Dr. Van-Hai Bui received his B.S. degree in Electrical Engineering from the Hanoi University of Science and Technology in Vietnam in 2013 and his Ph.D. degree from Incheon National University in South Korea in 2020. From 2021 to 2022, he was a Research Investigator and Lecturer in the Department of Electrical and Computer Engineering at the University of Michigan-Dearborn. Currently, he is an Assistant Professor in the School of Engineering, Department of Electrical Engineering at the State University of New York (SUNY), Maritime College. His research interests include energy management systems, applications of reinforcement learning in power and energy systems, and microgrid operations.

TUNING THE PARAMETERS: A MARITIME-TUNED MACHINE LEARNING COURSE

0: Abstract

In machine learning (ML) education, the choice of which datasets to utilize for student assignments and projects is critical for student success and meeting course learning outcomes. Poorly chosen datasets leave students disinterested and questioning the applicability of ML in real-world situations specific to their intended endeavors post academia. Additionally, some datasets demand much effort for preprocessing and a steep learning curve for understanding, which detracts from the ML experience and leaves students frustrated. As maritime and marine engineering programs expand to include ML in their curricula with the plan of addressing industry trends in, for example, autonomy and defense, it is important to calibrate the ML course accordingly with relevant datasets and assignments.

We develop a maritime-specific course in undergraduate ML (taken in the sixth semester) to engage students whose interests include maritime and marine industries. In support of our course, we compose several maritime-specific ML mini projects employing the popular and convenient Google Colab platform and make them publicly available within a GitHub repository. A hybrid of programming and report writing, each mini project utilizes the same publicly available maritime-related dataset—one that requires little preprocessing and, we show, is conducive for demonstrating many of the concepts vital to classical ML as well as some topics in deep learning. Using the same dataset for many assignments fosters a feeling of student comfortability, promotes comparing the performances of different ML algorithms, and provides a low barrier of entry after the initial assignment.

Our paper is both a detailed syllabus of a first course in maritime-focused ML and a how-to guide for effective use of the mini projects we have developed. Going further, it is a solution to the mini projects, as it reports on ML algorithms' performances, how the choices of key tuning parameters affect said performances, and how and why algorithms perform the way they do. Included in the paper is a student reflection authored by a US Coast Guard license student in engineering to offer instructors a unique student perspective and insight into the efficacy of the course design. Our hope is that colleagues interested in teaching a similar course at their own institutions can adopt our methods, and thereby reduce their preparation work and increase student engagement.

1: Introduction

1.0: Motivation

ML is becoming an essential component of the modern, evolving maritime industry, with use-cases including autonomous navigation, ship maintenance and monitoring, voyage optimization, ship design, and smart utilization of onboard electrical power distribution systems [1], [2]. With the industry expected to spend almost three billion dollars between 2022 and 2027 on ML

solutions, ML scientists and engineers with domain-specific expertise and the ability to design, implement, and troubleshoot ML systems will soon be in high demand [1]. The time is now for maritime educational institutions to adapt accordingly to maintain relevance and continue graduating in-demand maritime professionals.

In this paper, we present an undergraduate maritime-focused course in ML. The main component of our course, and the major contribution of this paper, is our design of several maritime-specific mini coding projects for addressing course learning objectives and ABET criteria, as well as engaging students. The following three components comprise the programming environment of each mini project:

- Colab: A product of Google that allows users to write and execute Python code through the browser, integrating it with a word processor. It also allows access to GPU to train the ML models and is well-suited for educational purposes.
- GitHub: A code-hosting and version control platform that allows users to work together on projects from anywhere and keep track of contributions between users.
- Google Drive: Cloud storage providing up to 15 GB at no cost. It allows users to store, synchronize, and share files across devices.

Figure 1 demonstrates the interaction among the three products. By making our projects publicly available online through GitHub, it is our goal that course designers at other institutions looking to implement a similar maritime-focused ML course may ease their workload by adopting our projects, which we describe in detail in Section 3. Also, for the purposes of this course, there is no barrier of entry for the use of the GitHub repository. Its only use in the course is to download the dataset and mini projects.

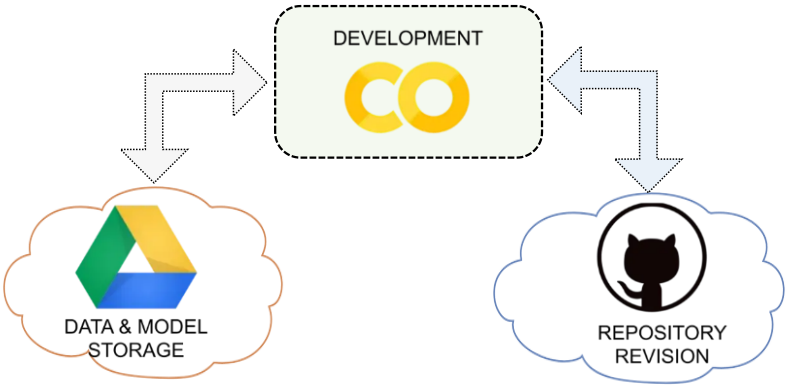


Figure 1. Environment setup.

A critical component to any ML project is data. Too much, too little, unclean, poorly documented/formatted, or uninteresting data may sabotage the student learning experience, leaving them disinterested, frustrated, or spending too much time up front in tedious preprocessing work [3]. Each of our five mini projects employs the *Ships in Satellite Imagery* dataset retrieved from Kaggle [4], which is a clean, well-documented dataset of reasonable size consisting of ship/no-ship colored images in the San Francisco Bay. The dataset, we show, goes a long way for demonstrating essential ML concepts such as classification, clustering, dimension

reduction, model assessment, and deep learning. We convert the original dataset into a Python-friendly format ("pickle" file), which we also make available on GitHub, and we include a black-and-white (B&W) dataset as well. Our students find this maritime dataset and associated mini projects engaging, which we support in Section 4 with a student reflection penned by a USCG License student in the SUNY Maritime College electrical engineering program.

1.1: Data Description

The original dataset includes 4000 red-green-blue (RGB) images 80-by-80 in size, 1000 of which contain a single whole ship. The 3000 others contain either no ship or a ship that extends off the image grid ("partial ship"). The dataset also includes the labels, '1' for ship, and '0' for no/partial ship. The B&W dataset is easier to study since B&W images consist of only a single channel. Figure 2 shows selected B&W and RGB images from the datasets.

Figure 2. (Left to right) B&W whole ship, B&W no-ship, B&W partial ship, RGB whole ship.

Additionally, we include a dataset that is the B&W dataset reduced to 300 dimensions (using principal component analysis, PCA) from 6400, which we provide to students to facilitate the train-testing of algorithms prior to studying dimension reduction. Working with dimensionally reduced image data is necessary to prevent overfitting and reduce computational effort. All our Python-ready datasets can be retrieved from the course repository. Students are instructed to download the datasets and subsequently upload them into their own Google Drive account, available for free.

1.2: Mini Projects Overview

The five mini projects we propose, summarized in Figure 3, each utilize the ship dataset and together cover the topics of logistic regression, model assessment, PCA, *K*-means clustering, and *K*-nearest neighbors. Students complete them in Google Colab, which is a convenient and free tool for integrating word processing with Python coding. Students are provided one Colab notebook (a file) for each mini project, with the notebook containing skeleton code and directions that can be very specific, or very general, depending on the task. The notebook, with its Python engine, will display plots, figures, images, and metrics. In addition to code, the student

can insert text into the notebook for qualitative responses. When completed, the notebook is easily converted to a report via print-to-PDF.

Google Colab and Drive play nicely with each other. The pickle data stored in Drive may be loaded into a Colab notebook with the Python code in Figure 4, which also shows manual permission will be required. Each mini project provides students with this completed code, partitioned into two sequential "cells"—see Figure 4. Upon running the first, students are prompted to enter their Google credentials to permit connecting Colab to Drive. A successful connection prompts the message *Mounted at /content/drive*. Running the second cell loads the ship data and prints the shape of the dataset, which confirms to the user that the data indeed loaded successfully.

After loading the data, the student completes the skeleton code and provides text responses as directed by the project. Each project addresses essential concepts and finishes with a *Go Further* section, possibly for extra credit, that pushes the students to complete an extension of the essentials. The projects employ a mix of *sklearn* pre-coded functions (i.e., "black-box" functions) with from-scratch coding, so students gain an awareness of the tools of the trade as well as develop an understanding of algorithm details. The notebooks containing Mini projects are provided in the course repository. The solutions are privately available by emailing Dr. Paul M. Kump at martymac13@gmail.com.

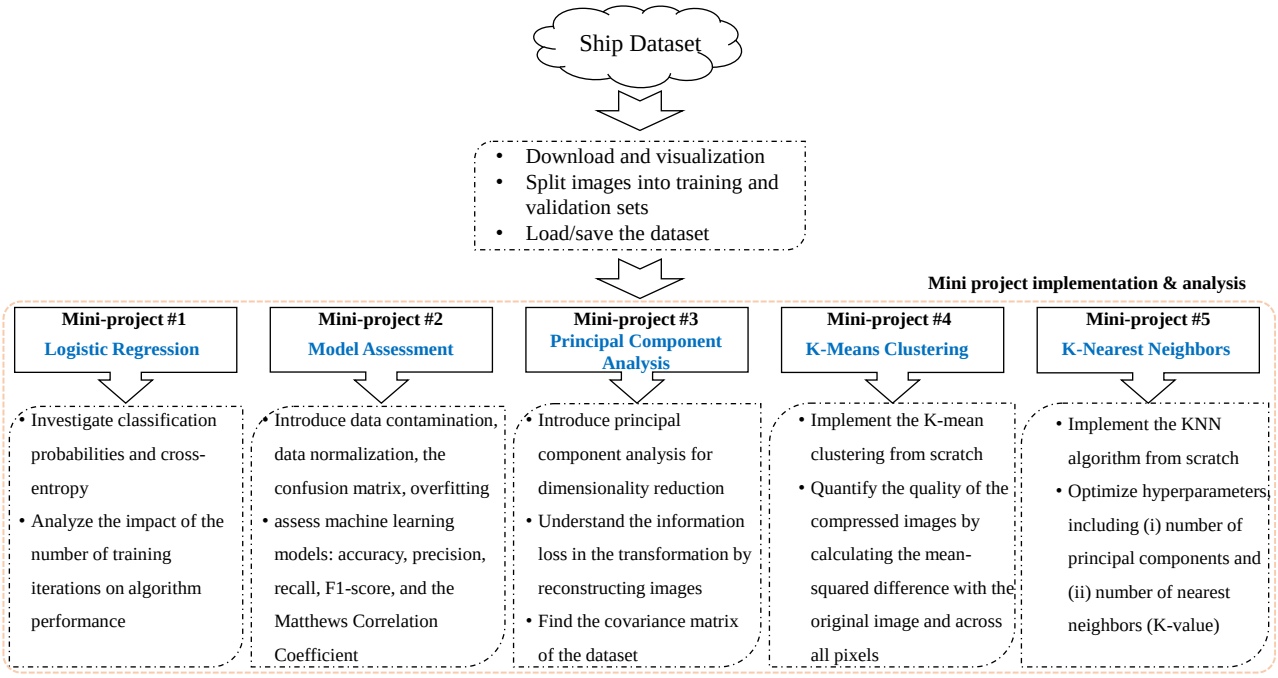


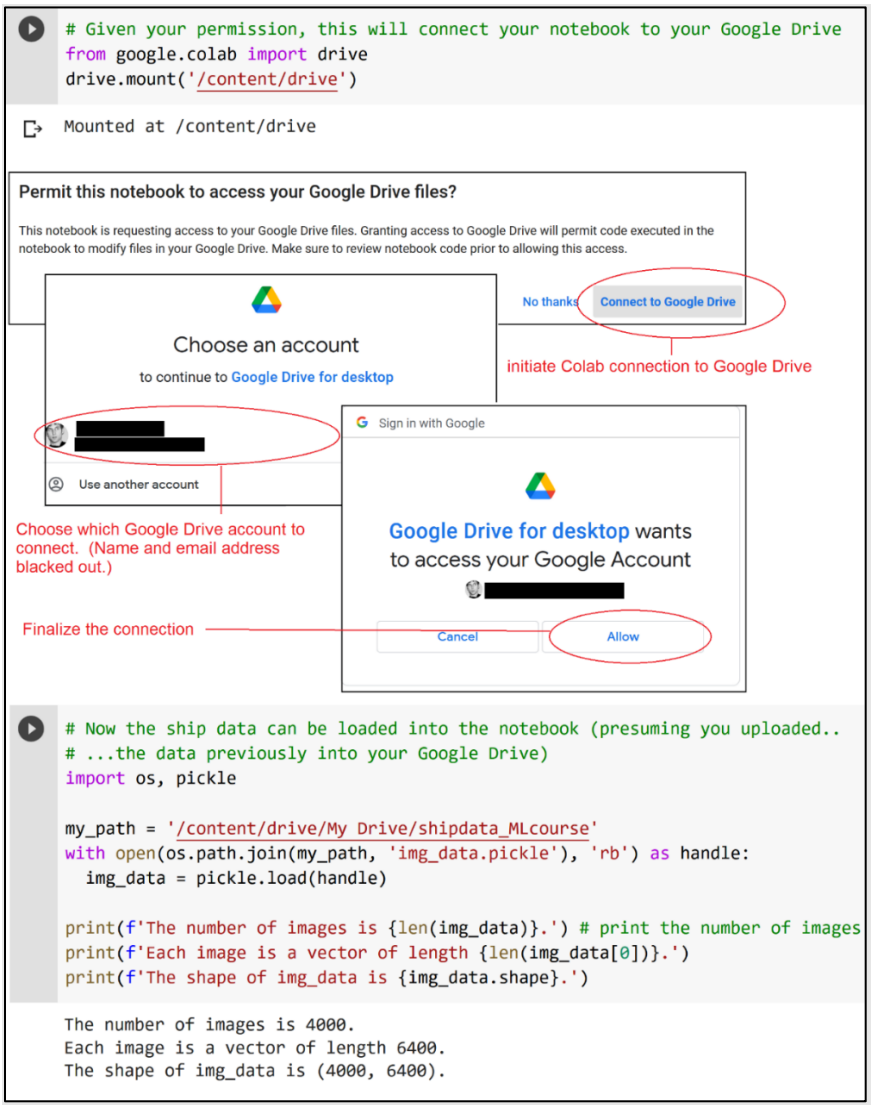
Figure 3. Summary of the five mini projects each utilizing the ship dataset.

2: Course Syllabus and Description

Our ML course is a three-credit upper-division course usually taken in the sixth semester by EE students at SUNY Maritime College, who need the course to graduate. The three credits are

incorporated as the equivalent of three lecture hours per week. Additionally, the course is popular among the whole engineering student body and typically draws students from other disciplines including marine engineering, mechanical engineering, and naval architecture—all of whom take the course for technical elective credit. About two-thirds of all students who take the ML course are license students.

Students are required to have satisfactorily completed a prerequisite programming course, which typically employs Python to demonstrate basic programming constructs like functions, loops, conditional statements, and data types. This prerequisite course is vital to our proposed ML course and schedule, which does not allocate time for covering programming basics. Students who transfer into the program with a language other than Python usually have little-to-no issues learning Python syntax quickly, which is facilitated by many professor-led Python examples.




```
# Given your permission, this will connect your notebook to your Google Drive
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
# Now the ship data can be loaded into the notebook (presuming you uploaded..
# ...the data previously into your Google Drive)
import os, pickle

my_path = '/content/drive/My Drive/shipdata_MLcourse'
with open(os.path.join(my_path, 'img_data.pickle'), 'rb') as handle:
  img_data = pickle.load(handle)

print(f'The number of images is {len(img_data)}.') # print the number of images
print(f'Each image is a vector of length {len(img_data[0])}.')
print(f'The shape of img_data is {img_data.shape}.')
```

```
The number of images is 4000.
Each image is a vector of length 6400.
The shape of img_data is (4000, 6400).
```

Figure 4. Connecting Google Drive and Colab accounts, then loading in the data.

A course in engineering statistics is also a prerequisite, though less important for successful completion of the ML course. It is reasonable to expect students to complete the projects without

having extensive statistical knowledge. Further, students beginning the course will already have learned in their math courses to perform simple matrix and vector operations like addition, subtraction, multiplying, transposing, and inversing; however, a rigorous course in linear algebra is not required nor expected. Instead, the proposed ML course allocates time to cherry-pick useful topics and theory from linear algebra in the context of ML on an as-needed basis. Students enjoy this because they see the theory on the board and then immediately apply it with programming.

2.0: Learning Outcomes

The mini projects we propose address at least three of the four (in-house) course learning objectives, stated as follows:

Upon successful completion of the course, the student will

- *(Objective I.) Gain in-depth knowledge of the basic principles of ML and its core algorithms. The student will gain an awareness of ML systems that are ubiquitous in daily life and understand why personal data are so valuable in modern intelligent computer systems.*
- *(Objective II.) Improve programming skills and become proficient in the Python programming language.*
- *(Objective III.) Understand key concepts in the subject of linear algebra and how they apply to computing.*
- *(Objective IV.) Gain engineering experience functioning in a group setting.*

Naturally, a student completing the Python-based mini projects will enjoy improved programming skills *(Objective II.)* Each mini project is strategically written to emphasize the basic principles governing core ML algorithms, which partially addresses *(Objective I.)* For example, the logistic regression project allows students to observe the consequences of data that are high-dimensional, and how the number of training iterations influences algorithm performance. Moreover, mini projects leverage the report-programming hybrid format of Google Colab to remind the student of important results from linear algebra and connect them to programming, thereby promoting *Objective III.* Mini projects should be completed by each individual student, it is recommended, and they do not address *Objective IV.*

Additionally, the mini projects address three of the four course ABET criteria:

- *(Criterion 1) an ability to identify, formulate and solve complex engineering problems by applying principles of engineering, science, and mathematics.*
- *(Criterion 5) an ability to function effectively on a team whose members together provide leadership, create a collaborative and inclusive environment, establish goals, plan tasks, and meet objectives.*
- *(Criterion 6) an ability to develop and conduct appropriate experimentation, analyze, and interpret data, and use engineering judgement to draw conclusions.*
- *(Criterion 7) an ability to acquire new knowledge as needed, using appropriate learning strategies.*

Every mini project requires that students identify, formulate, and solve complex engineering problems (*Criterion 1)*, and no mini project has a single, unique solution, which therefore

emphasizes the complex nature of the problem. The supplied code and comments in each mini project provide students with only a broad goal; they must identify, formulate, and solve the problem. The instructor has the freedom to decide what milestones satisfy these three components. Regarding *Criterion 6*, mini projects generate many plots, pictures, and graphs requiring students interpret results, summarize, and draw conclusions. Moreover, mini projects address *Criterion 7* by often introducing a new programming function or method requiring students to consult online documentation for successful implementation. Again, the mini project's report-programming hybrid format enables links to be embedded in the assignment to point the student in the right direction. *Criterion 5* is not addressed by the mini projects.

2.1: Topics and Schedule

General topics covered in our course are the same as any non-maritime ML course to ensure students learn the same necessary fundamental concepts. These topics are regression, classification, clustering, dimension reduction, model evaluation, and bias-variance tradeoff/overfitting. Influencing the course schedule is a key assessment unrelated to the mini projects: a group project designed by the students and in lieu of a final exam. The schedule is carefully designed to provide students with a holistic view of ML early in the semester to maximize the number of options available to students when choosing their project and promote a variety of student projects. Students are required to settle on a topic about 10 weeks into the 15-week semester, as shown in the schedule in Table 1.

Table 1. Course schedule.

Week	**Topics**	**Related Mini Projects**
1 - 4	ML overview, matrix review, linear regression	None
5 - 7	Logistic regression, model assessment	Mini Projects #1, #2
8, 9	Principal component analysis, *K*-means clustering	Mini Projects #3, #4
10	*K*-nearest neighbors	Mini Project #5
11-14	Optional material, e.g., deep learning, random forest, linear discriminant analysis	None. Students work on designing and completing their final project.
15	Finals week	None. Students present their original project to the class

The first four weeks include an overview of ML, a review of matrix operations, and linear regression. After this time, students should understand ML at a high level, how it is applied, and the aim of different types of algorithms (e.g., classification, clustering). Students realize that matrices and data are intimately related and can program simple matrix operations and observe the results. They understand and can program the nuts-and-bolts details of linear regression, including the normal equation (i.e., ordinary least squares) and gradient descent to determine model parameters. Any classroom demonstrations during this time are conducted in Google Colab so the environment becomes familiar to the students.

The next three weeks cover both logistic regression (Mini Project #1) and model assessment (Mini Project #2), the latter of which investigates bias-variance tradeoff. During this time,

students are introduced to classification. Having previously completed regression, students therefore obtain exposure to the two most prominent types of ML algorithms and how to evaluate them, all in the first seven weeks of the course.

Unsupervised learning is covered in Weeks 8 and 9. During this time, we suggest covering PCA (Mini Project #3) as soon as possible, since all the mini projects utilize PCA-reduced data, even Mini Projects #1 and #2. This period will naturally include discussions on covariance, eigenvectors, and projection, for which the generous two-week period allows. It may seem strange (to the instructor) to cover PCA after PCA-reduced data are provided to the students in the first two mini projects, but we believe the alternative should be avoided since otherwise students would not have a good understanding of features and the drawback of having too many features—and thus cannot appreciate PCA. We elaborate on this issue in Section 3 when we describe the mini projects in detail. With K-means clustering immediately following PCA, students apply both algorithms in tandem to visualize the clusters in a reduced two- or three-dimensional space (Mini Project #4).

Covering *K*-nearest neighbors (KNN), Week 10 is a milestone week since it is what we deem as the last of the necessary material. Studying this simple algorithm last invites easy integration of a classifier with both PCA and model assessment to provide students experience with a bona fide ML pipeline (Mini Project #5):

Original data -> PCA -> Classifier -> Assessment

Week 10 is also a good week for a mid-term exam, since KNN, as simple as it is, does not warrant a full week for understanding. At the end of this week, students are expected to understand all the essentials of ML and are in a great position to choose a final project, if instructors choose to go that route.

3: Mini Projects Details

3.0: Mini Project #1: Logistic Regression

Presumably the students' first exposure to the ship dataset, classification, and computer vision, this assignment requires that students explore the dataset and use black-box logistic regression to classify images as ship/no-ship. Students are provided with the original image dataset to view the images and a dimensionally reduced dataset (from PCA) to facilitate algorithm training. As logistic regression is a supervised learning algorithm, the assignment also provides students with the target dataset for training and observing accuracy. Classification probabilities and cross-entropy are investigated, as are model parameters. The *Go Further* section encourages experimentation with changing the number of training iterations and the consequences of using the original image dataset instead of the dimensionally reduced one.

A dimensionally reduced dataset is necessary for the logistic regression algorithm to converge. However, students have no prior knowledge of PCA analysis or any other dimension-reduction algorithm at the point of completing this mini project, and therefore cannot yet understand the reduced dataset. To keep the assignment focused and simple, it provides students with only brief, general information regarding dimension reduction. This is not a significant hurdle, though, and even promotes curiosity, as well as provides an opportunity for comparison. When students

complete Mini project #3 - Principal Component Analysis, performing the dimension reduction for themselves, they may compare their results with the reduced dataset they are provided.

Evaluating the model is kept simple to maintain focus on logistic regression. The assignment simply requires that students compute the rate of correct predictions (accuracy) and the model's cross-entropy—a standard performance metric for logistic regression—on the entire dataset instead of splitting it into training and testing partitions. A separate mini project rigorously addresses model evaluation. We find the logistic regression model makes 3678/4000 (92.0%) correct predictions with a cross-entropy of 2.78. The assignment places the latter into context when it requires students to compare the cross-entropy of a trivial model making random predictions, which we find to be 17.1. Students are expected to provide insight that the learned model minimizes cross-entropy, so it makes sense that the observed cross-entropy of a trained model is much less than that of the random model. Requiring additional insight, the assignment asks students to find one image whose class probability is just above 50% and describing what they see in the image. We find the model correctly assigns the image in Figure 5 to the no-ship class, but only with a small probability of 0.54, which is most likely due to the long, white linear feature resembling a ship.

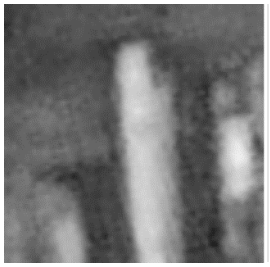


Figure 5. Logistic regression predicts this image as no-ship with relatively small probability.

3.1: Mini Project #2: Model Assessment

In this mini project, students gain a robust view of core machine learning concepts, including data contamination, data normalization, the confusion matrix, overfitting, and the basic performance metrics for predictive models. Unlike the previous mini project, where the objective is to familiarize students with the details of logistic regression and the dataset itself, students begin to understand the additional steps necessary before and after training a machine learning model. Upon completion of this mini project, students have exposure to the full machine learning pipeline, which is used and elaborated upon throughout the course with different algorithms. This mini project begins in the same manner as the previous, with a brief exercise of dataset indexing and visualization. Then, students are again provided with the dimensionally reduced dataset and a brief explanation as to why this step is necessary. As mentioned, this step is the topic of focus in Mini Project #3.

The first important concept that is addressed is the problem of data contamination. Understanding that it is necessary to split the dataset into a training set and a testing set to avoid misleading results, students are prompted to use a black-box function to perform the split. Immediately following this operation, students are asked to normalize the training set and testing

set to zero mean and unit variance, again using a black-box function. The key takeaway from this step is that the testing set must be normalized in the same manner as the training set to ensure that predictions on the testing data sufficiently represent model performance. Understanding that the data preparation steps have been completed, students then train a logistic regression model and obtain predictions on the training set and testing set. Later in the mini project, the concept of K-Fold Cross-Validation is introduced, which adds another method of reducing data contamination into the students' arsenal. Students see that an anomalous testing set can have a similar effect on the results as an anomalous training set. Therefore, they are prompted to use the black-box cross-validation function to train the model K times with different training and testing sets. They then take the mean and standard deviation of the accuracies from each of the K folds, obtaining values that are better estimates of model performance.

The next segment of this mini project dives into some of the key performance metrics needed to assess machine learning models: accuracy, precision, recall, F1-score, and the Matthews Correlation Coefficient. Using a black-box function to obtain the confusion matrix from both the training predictions and testing predictions, students use formulas to calculate these metrics for both sets and realize that the results are better for the training set. For the training set, the model calculates an accuracy of 92.8%, a precision of 0.891, a recall of 0.809, an F1-score of 0.848, and an MCC of 0.802. The testing set did not perform as well, with an accuracy of 87.0%, a precision of 0.746, a recall of 0.746, an F1-score of 0.746, and an MCC of 0.659. The purpose for calculating these metrics for both sets is to display this observation and stimulate thinking into why the predictions on the training set are better. They also realize that accuracy itself is not sufficient to fully assess a model. Therefore, the precision, recall, F1-score, and MCC must also be considered. This observation prompts students to discuss reasons for why this is true, such as class imbalance.

In the *Go Further* section, students are introduced to the concept of overfitting and its importance in model training. For Logistic Regression, the regularization hyperparameter (a.k.a. C-value) regulates overfitting. To display the effect of regularization on model accuracy for the training set and testing set, students are asked to plot the model accuracy versus C-value for logarithmically spaced values between 0.001 and 100. Figure 6 shows the results, emphasizing that decreased regularization (increased C-value) reduces overfitting and can improve model performance.

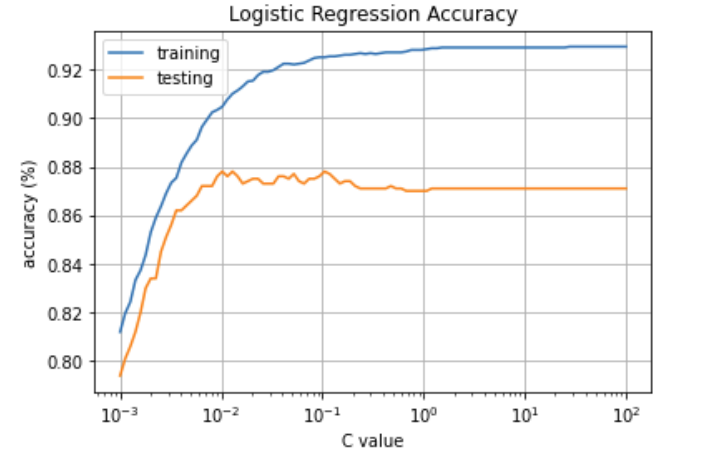


Figure 6. Model Accuracy vs. C-Value.

3.2: Mini Project #3: Principal Component Analysis

In the course's third mini project, students learn the details of PCA left out in previous assignments. While the students have used the dimensionally reduced dataset before, they have not yet learned how variance is retained in the dataset after the transformation. Therefore, this assignment's purpose is to elaborate on this step to give students a deeper understanding of the data compression portion of the machine learning pipeline. Through the visualization of the cumulative scree plot, students learn to choose how many principal components should be retained in the transformed dataset. To evaluate the performance of the from-scratch algorithm, the results are compared to those yielded by the black-box function. Further, the students visualize the information loss in the transformation by reconstructing the images and comparing them with images from the original dataset. This transitions to the other use of PCA, which is anomaly detection.

The assignment begins by providing the students with the original dataset containing the black and white images and their corresponding targets. After centering the data, the students are prompted to find the covariance matrix of the dataset. Knowing that the principal components are the eigenvectors of the covariance matrix, they proceed to calculate the eigenvalues and eigenvectors. After ordering the eigenvalues in descending order with their corresponding eigenvectors, students plot the cumulative scree plot, which indicates the number of principal components necessary to retain any given fraction of the variance in the dataset. From this plot (shown in Figure 7, top left), students understand that most of the variance in the original dataset is retained by few principal components. To further prove this discovery, they are then asked to find the fractional retained variance for the first 300 principal components, which is about 96%.

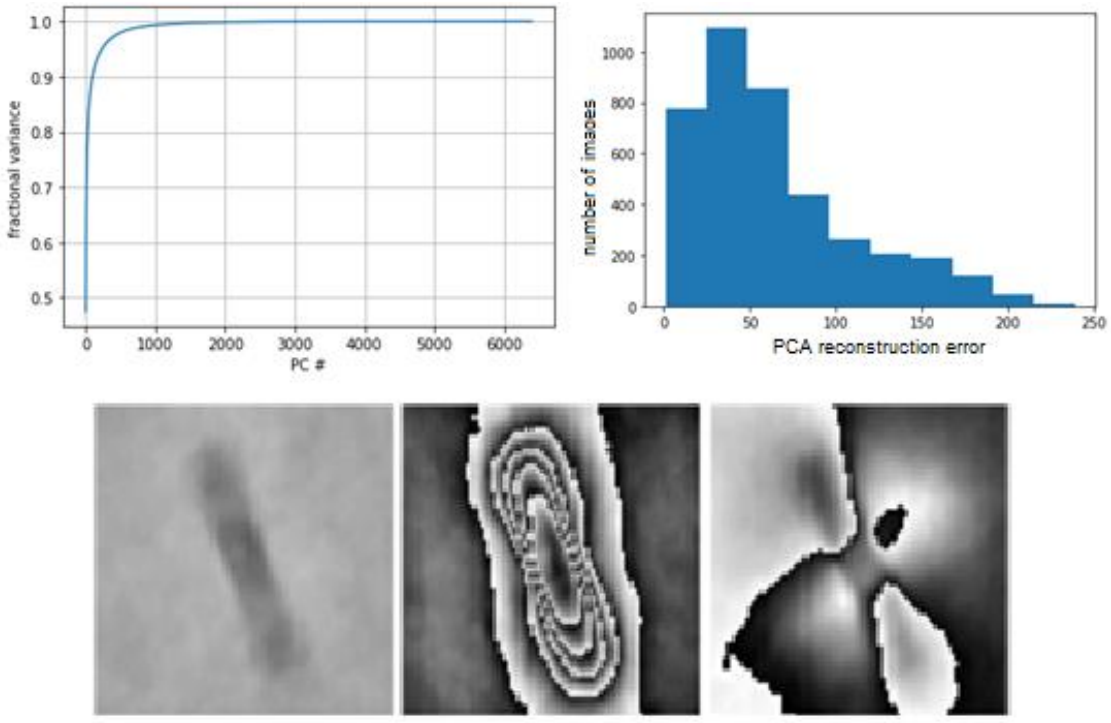


Figure 7. PCA analysis of B&W ship dataset.

After performing the transformation from 6400 principal components to only 300, students compare their results with the reduced dataset given to them in mini projects #1 and #2—one that was reduced using a black box function. Comparing individual elements in each array, they find that the differences are negligible, and the transformations are nearly identical. Then, to demonstrate that information is lost after transformation, the students are prompted to reconstruct the images by undoing the transformation and centering. Doing so allows for comparison between images in the original dataset and the reconstructed dataset through the Mean-Squared Error (MSE). Students then calculate the MSE and plot a histogram of the reconstruction error, which is shown in Figure 7, top-right.

The next section of the mini project exposes the students to the anomaly detection abilities of PCA. With the reconstruction error of the images known, students are asked to transform and reconstruct an anomalous image (from outside the dataset) to calculate its reconstruction error. They find that the error for this image is 815, which is much higher than any image from the dataset and therefore, it is an anomaly.

In the *Go Further* section, students visualize the sorted eigenvectors of the covariance matrix and make an interesting discovery. Since each eigenvector is an array of length 6400, students reshape and display them as images. They find that the first eigenvector closely resembles a ship-class image, and the subsequent eigenvectors become increasingly unstructured. This captures the importance of sorting the eigenvalues in descending order, such that the leading eigenvectors represent most of the variance in the dataset. Figure 7 (bottom row) shows the first three reshaped eigenvectors.

3.3: Mini Project #4: K-Means Clustering

This mini project addresses the subject of image compression, an important application of *K*-means clustering. Students are asked to code the algorithm from scratch and use it to determine the *K* color hues that best represent a single user-selected image from the original (RGB) ship dataset. Because pixels in any color image are each a three-dimensional vector of intensities, they can be imagined in three-dimensional space and clustered appropriately. Each of the *K* cluster centers represents an optimal color hue to which intra-clustered pixels may be quantized, thereby reducing the number of color hues comprising the image to *K*. The value of *K* is critical; as it decreases, the image is compressed more but less recognizable, retaining less information.

Students are initially instructed to set the value of *K* to 16 and then complete several lines of the provided code to program a working *K*-means clustering algorithm. They are asked to print the 16 means and should interpret the results in their written report. With the 16 means and the clustering information resulting from the algorithm, a subsequent block of code will display the compressed image, and students should interpret the image that results. To quantify the quality of the compressed image, students are then asked to calculate the mean-squared difference with the original image and across all pixels. Students are instructed to rerun the code with different values of *K* and each time note the means, observe the resulting image, and calculate the quantization error—a process that promotes understanding the algorithm and its results.

For practical reasons, the value of *K* is often a power of two, and for the ship dataset, should be no greater than 32. As the value of *K* decreases, the error increases as expected. We find that

with $K = 2$, the hues that result from running the algorithm are [R,G,B] = [174, 147, 126] and [60, 77, 64], which are the integer-rounded cluster means. The compressed image in Figure 8 (middle) illustrates the two hues; the ship is gray-blue, and the water swamp-green. From an RGB color slider available for free on the internet, one may deduce the gray-blue ship corresponds to the hue [174, 147, 126], and the swamp-green water [60, 77, 64]. We find the quantization error is 14.2.

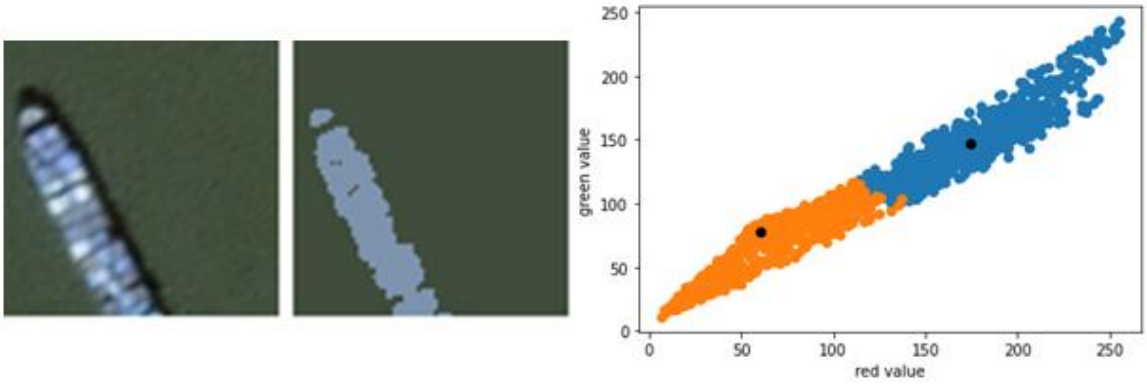


Figure 8. *K*-means clustering analysis of RGB ship dataset.

The penultimate directive is to ignore a channel, say the blue channel, and observe the clustering information in two-dimensional (red, green) space. Doing so both verifies the correctness of the algorithm's output and promotes a deeper understanding of the results. The code provided in the mini project will display the *K* clusters as *K* separate colors unrelated to RGB channels and the *K* means as filled black circles. Figure 8 (right) is an example of the code output when $K = 2$. An observant student should be able to point out that the black circles are not in the middle of the clusters, at least as shown in the figure. Students should be able to provide insight that more "mass" (pixels) is clustered around the means which is not immediately apparent from the figure. Further, not all the information is contained in the figure since the third dimension (e.g., blue channel) is not shown.

Since the original ship images are small and do not contain many unique color hues to begin with, it can be helpful for students to upload and process their own larger images rich in colors, which is what the *Go Further* section encourages. Interacting with the project via a custom image instead of one provided is a source of gratification for the students. Students are warned though that processing time on a large custom image may take several minutes, and the project recommends to first crop the images accordingly. An image rich in colors can be compressed even with large values of *K*, e.g., 512 or 256. The students enjoy observing the results of running their code on their own images.

3.4: Mini Project #5: K-Nearest Neighbors

Up to this point in the course, students have only been exposed to one classification algorithm—Logistic Regression. Now, they are introduced to *K*-Nearest Neighbors (KNN), which is another widely used shallow learning classification algorithm. Another focus of this mini project is

hyperparameter optimization, which is an essential part of model building. The two hyperparameters that dictate the results of this model are the number of principal components and the number of nearest neighbors considered (*K*-value). Using hyperparameters that are not optimized, students are asked to optimize them and compare the results to those yielded by logistic regression. This comparison encourages the students to explain the differences between the algorithms, strengthening their understanding of the details behind them. It also encourages students to be creative in optimizing the parameters, using methods such as trial-and-error or the more advanced grid search. Since hyperparameter optimization via trial-and-error is tedious, the students should use their knowledge of the python language to find a more efficient way to find the optimal parameters. In the *Go Further* section, they are asked to compare the results of the from-scratch KNN algorithm with the results of the black-box algorithm.

After using black-box functions to split, normalize, and perform PCA on the dataset, students are asked to code the KNN algorithm from scratch. The first time around, only two principal components are retained in the reduced dataset. To foreshadow the inadequacy of retaining these many principal components, the dimensionally reduced dataset is plotted in two dimensions with color-coded classes. Figure 9 shows this plot, which displays that there is minimal split between classes in two-dimensional space.

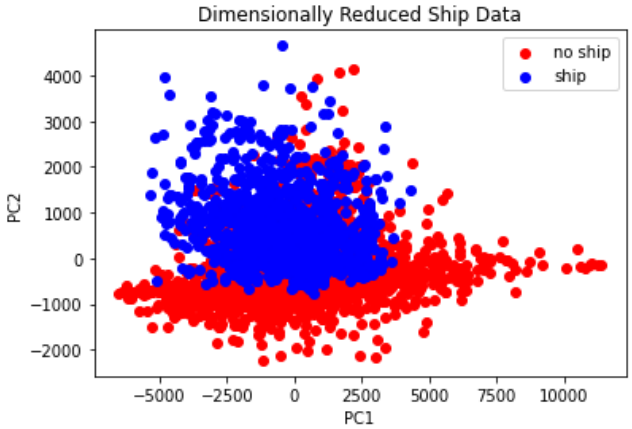


Figure 9. Class split with two principal components.

The students find that the results are poor with these hyperparameters, which yield an MCC of 0.336, an F1-score of 0.490, and an accuracy of 75.9%. The black-box function yields the same results as the from-scratch algorithm. Students are then asked to improve the model results by varying the number of principal components retained and the *K*-value. If they perform this step correctly, they find that the optimal hyperparameters for this model are about 100 principal components with a *K*-value of 5. With these hyperparameters, the testing predictions yield an MCC of 0.898, an F1-score of 0.924, and an accuracy of 96.1%. Comparing their results to those yielded by logistic regression in Mini Project #2, students find that KNN performs substantially better.

3.5: Deep Learning Considerations

Seeing the limitations of shallow learning methods in classifying the dataset's images, it is interesting to consider the performance of more advanced methods such as deep learning—more

specifically transfer learning. At the course's end, students may be interested in seeing the next level of the subject. Therefore, we implement transfer learning on the dataset using the GoogLeNet [5] architecture—a 22-layer convolutional neural network that is pre-trained with the ImageNet dataset [6]. Given that our dataset is small and insufficient to train an entire neural network, this method uses the pre-trained weights embedded within the architecture to recognize features within the dataset, essentially retraining only the surface layers of the network.

Using this method, we are pleased to see, without surprise, that we obtain better results than any shallow learning method. Table 2 shows the results of transfer learning for the training set and the testing set. Here, we see that the model performs better in the training step for both the black & white and the color images. We also see that the model performs better with the color images in both steps, which makes sense because the ImageNet dataset comprises color images with RGB channels.

Table 2. Deep learning classification results on the testing set and the training set (in parentheses).

	B&W Images	RGB Images
Accuracy	98.7% (99.3%)	99.1% (99.8%)
Precision	0.967 (0.988)	0.976 (0.996)
Recall	0.983 (0.987)	0.988 (0.997)
F-1 score	0.975 (0.987)	0.982 (0.997)
MCC	0.968 (0.983)	0.976 (0.996)

4: Student Reflection

The following is a reflection of the course written by an electrical engineering student in the USCG License program at SUNY Maritime College:

For students to fully grasp a subject, it is imperative that the subject grasps their interest. I am interested in ships and other maritime-related subjects. This course offers a way for students like me to learn the basic ML concepts in such a way that sparks our interest. Using *Ships in Satellite Imagery* dataset, we see that ML has applications in our industry, and that with our knowledge obtained from the course, we can bring skills to the table that are both unique and sought after by employers. Professors that seek to use our methods can easily adapt the course to the interests of their own students, choosing a dataset that fits their interests more closely.

While the content of the dataset is imperative in catching the attention of students, it is also important that students can easily access it, which is the first step towards using it in machine learning applications. For students who have not been exposed to ML previously, it may be difficult for them to get started due to difficulties in importing the data, which may discourage them and take away from their experience in the course. This problem was solved by using Google Colab, which interfaces with Google Drive. By simply importing the dataset to Google Drive, students can easily connect the two platforms and import the data with only a few lines of code. Since the dataset does not change throughout the course, the process does not change either. Therefore, students will not be bogged down with data processing steps and can put all their focus on ML concepts. This is just one of the many reasons that Google Colab is a very suitable platform for this course. From a student’s perspective, this platform offers a clean and

easy-to-use interface. From a professor's perspective, assignments can be creatively organized in blocks, ensuring that students can clearly identify the objectives of the assignment and satisfy them accordingly. With clear instructions and a clean environment, there is abundant potential for student success.

Another important component in ensuring that students understand the course material is the ordering of the assignments. With the content ordered in such a way that provokes curiosity, students will be engaged and looking forward to learning the details of content that was mentioned but not fully elaborated upon early in the course. For example, after loading in the original dataset in Mini Project #1, students are told that they will be using the dimensionally reduced dataset instead with a brief explanation of the purpose of PCA. They are then exposed to the concept of data normalization, which is also elaborated upon later in the course. While the purpose of Mini Project #1 is to dive deep into the details of logistic regression, it also raises questions about the ML pipeline. This project also keeps the model assessment portion of the pipeline simple such that the focus is kept on the logistic regression classifier.

Moving into the next assignment, students know the very basics of the pipeline from loading in the data to obtaining a model assessment metric (accuracy). Now that the students have seen this, instructors can be confident that adding extra steps/concepts will not cause overwhelming confusion. Therefore, the next assignment in the course adds the step of splitting the dataset into a training set and a testing set as well as the concept of *K*-fold cross validation. It also introduces the other model assessment metrics besides accuracy, which are precision, recall, F1-score, and the MCC. Again, to foreshadow what is coming later in the course, the *Go Further* section introduces hyperparameter optimization in logistic regression. Students who are ambitious and interested in completing this section will benefit and have an edge over those who only do what is required.

The third mini project answers the questions about PCA that students were waiting for. Here, they learn why the original dataset is not used and how data compression is performed. At this point in the course, students should be more comfortable with python coding. Since this assignment is more programmatically intensive than the previous, it is better that it is placed towards the middle of the course. Upon completion of this assignment, students have visual representations of their algorithm's performance through the cumulative scree plot and reconstructed images. Going forward in the course, they can utilize these skills to improve their models and prove their understanding of the content. Not only does this assignment introduce the data compression capabilities of PCA, but it also shows how it can be used as an anomaly detector. This practical application is remarkably interesting and proves to students yet again that machine learning is an invaluable tool.

After learning about PCA, students are introduced to a different method of data compression. This method is *K*-Means Clustering. Rather than reducing the dimensions of the dataset, they reduce the number of color hues within the images. Understanding that as the number of hues goes down, the image becomes less detailed, students are asked to experiment with the *K* hyperparameter and view the quantization error between the original image and the reduced image. From past assignments, students understand that the *K*-value is a hyperparameter, and to use this algorithm, they must optimize it with an acceptable quantization error. In the *Go Further* section, students are asked to upload their own image and reduce the number of hues it contains, further showing the applicability of this algorithm.

The purpose of the *K*-Nearest Neighbors mini project is to add another classification algorithm to their arsenal. Noticing that they have the knowledge and skills to learn more algorithms, students realize that they do not need to stop at just two classifiers. If they choose, they can take it upon themselves to use the pipeline that they are acquainted with to expand their knowledge of the subject. This assignment also focuses heavily on hyperparameter optimization with not one, but two hyperparameters. Now, students can creatively find practical ways to optimize their models, which is an invaluable skill.

From a student's perspective, this course is organized in a way that promotes a higher level of understanding of ML. Through completion of these assignments, students learn the applicability, uniqueness, and innovativeness of this subject, which may motivate them to learn more and incorporate it into their future endeavors. Personally, the subject of ML has influenced my career trajectory going forward, and I hope that this course influences the lives of other students as much as it did for me.

5: Conclusion

This paper presented a maritime-focused course in ML, emphasizing five original mini projects that utilize the *Ships in Satellite Imagery* dataset to demonstrate essential ML concepts such as classification, clustering, dimension reduction, and overfitting/model assessment. Leveraging popular, free and robust cloud-based tools, our projects and data are easily accessible for students, encourage a hybrid of report writing and Python coding, and are publicly available to all course designers. Using the same maritime-focused dataset promotes deeper understanding and limits frustration, at least for US Coast Guard license students.

There are, of course, some limitations from using the same dataset for every project, particularly with the *Ships in Satellite Imagery* dataset. One is that it is difficult to draft meaningful *regression* projects with the ship dataset—or any image dataset. Thus, some key topics in ML like linear, lasso, and ridge regression may necessitate a different dataset. It is our future goal to experiment with other maritime-related datasets to find a single dataset conducive to regression analysis. A second concern is that students may grow bored from analyzing the same dataset, though there is no evidence at this time to support that idea.

Like the methods in [7], we plan to compare the level of achievement of ABET and course learning outcomes before and after these mini projects are deployed into the ML course. Following [7], we also plan to survey the students with targeted questions to precisely understand the qualitative effect, if any, our mini projects have on the students. Though this paper included a student reflection of the course, data from many more students is needed to accurately judge the impact and efficacy of the course.

6: References

[1] Offshore Energy, “Maritime industry to spend $931 mln on AI solutions in 2022,” 20 January 2023. [Online]. Available: https://www.offshore-energy.biz/maritime-industry-to-spend-931-mln-on-ai-solutions-in-2022/#:~:text=There%20is%20a%20wide%20range,maintenance%20and%20monitoring%20of%20vessels.

[2] A. Grech La Rosa, E. Anderlini and G. Thomas. (2021) “Preliminary bulbous bow design tool applying k nearest neighbours classification and regression model,” *International Journal of Maritime Engineering*, vol. 163, no. A3, https://doi.org/10.5750/ijme.v163iA3.806

[3] N. Lavesson. (2010) “Learning machine learning: A case study,” *IEEE Transactions on Education*, Vol. 43, No. 4, pp. 672-676.

[4] “Ships in satellite imagery: Classify ships in San Francisco bay using planet satellite imagery,” [Online]. Available: https://www.kaggle.com/datasets/rhammell/ships-in-satellite-imagery,Accessed 24 September 2022.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. (2015) “Going deeper with convolutions,” *IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.

[6] J. Deng, W. Dong, R. Socher, L. -J. Li, Kai Li, and Li Fei-Fei. (2009) "ImageNet: A large-scale hierarchical image database," *IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[7] P.M. Kump. (2021) “How classroom flipping affects coast guard license students in engineering,” *2021 ASEE Annual Conference*, Virtual meeting, https://peer.asee.org/how-classroom-flipping-affects-coast-guard-license-students-in-engineering