Assignment 4: Machine Learning

In this assignment, we will explore the fundamentals of machine learning with a hands-on project using Teachable Machine, a web-based platform designed to simplify the creation and training of machine learning models. We will experience the complete lifecycle of a machine learning project focused on solving a classification problem—from preparing and organizing the dataset to training the model, evaluating the model, and analyzing how variations in data distribution can impact the model performance.

Part 1: Machine Learning Foundations

Classification problem

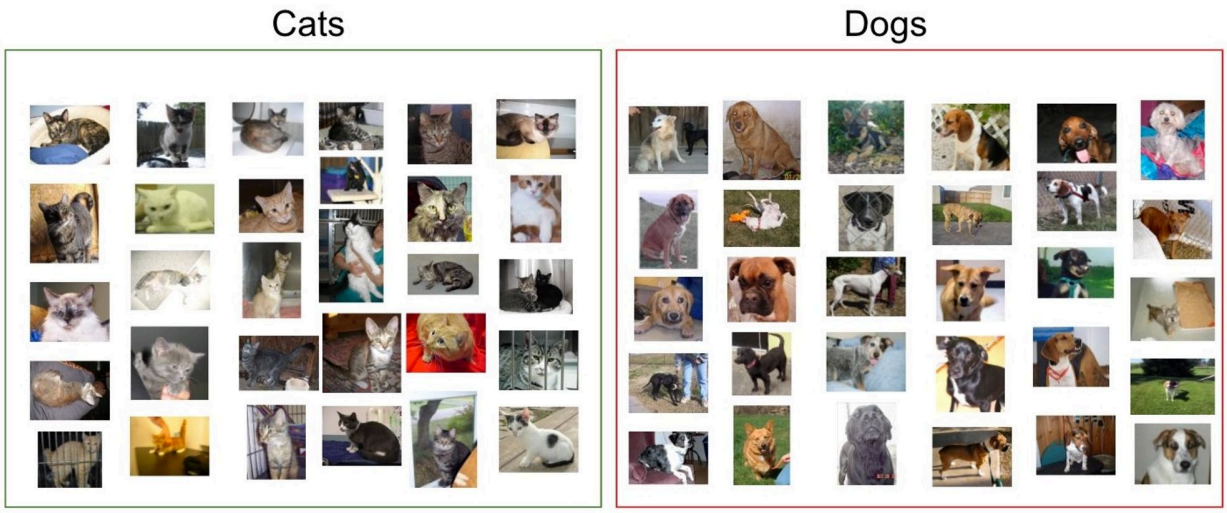
This part covers several fundamental concepts that underpin machine learning. We will start with the concept of classification. The objective of a classification problem is to decide the categorical label for an input object. Examples include identifying whether an email is spam or not, or classifying images of animals by species. Classification problems can be categorized based on the number of labels they involve: those with only two possible labels are known as **binary** classification problems, while those with more than two labels are referred to as **multiclass** classification problems. Below are examples of binary and multiple classification problems:

- **Binary:** given a picture of water bottle, determine if the bottle is filled or empty
- **Multiclass:** given a picture of a computer, determine if the computer is a laptop, desktop, or all-in-one.

Try to identify two classification problems you might encounter in daily life, one binary and one multiclass:
[Student answer:
]

Training data

Machine learning is a data-driven approach, meaning that it does not have any prior knowledge or rules for solving the problem. Instead, it learns from the data to solve the problems. Taking a typical classification example: given an input image, determine it to be either a “dog” or a “cat.” We humans have the solution to this simple problem because we have seen how cats and dogs look in our life experience. For example, dogs usually have more pronounced muzzles, while cats usually have shorter, flatter faces with pointed ears that are typically upright and more triangular in shape. A machine learning model solves the problem similarly. It is given a set of training data that looks like this:



Each **training data point** in this dataset is an image associated with a label of either “dog” or “cat.” By training on this dataset, a machine learning model would likely determine that an image similar to the right pictures is a “cat,” and an image similar to the left pictures is a “dog.”

Now, imagine that we swap all “dog” and “cat” labels in the dataset. By training on this new dataset, what would the machine learning model be likely to predict for a dog image?

a. Predict “dog” label because the image shows an actual dog;
b. Predict “cat” label because the image is similar to pictures with “cat” labels in the dataset.

[Students answer a multi-choice question. (b) is the correct answer]

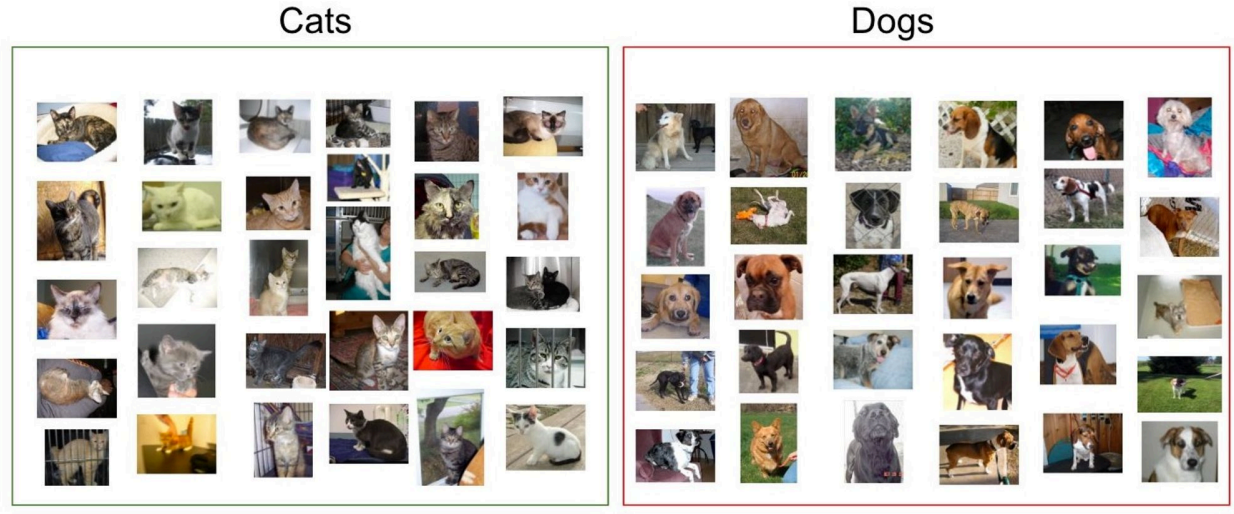
Generalization gap

Imagine a child who has just started learning elementary arithmetic. He might practice problems like 1+1=2 or 2×3=6. However, during an exam, we usually test his understanding with different numbers than those he practiced with. The difference between his performance on the exam and his practice is known as the generalization gap.

Similarly, in machine learning, it is crucial for a model to make accurate predictions on data that was not part of its training. If we refer to the data the model needs to predict in the future as the **test dataset**, then the **generalization gap** is the *difference between the model's performance on the training dataset and its performance on the test dataset*. This gap tends to be larger when the test data is significantly different from the training data. For example, if a machine learning model is trained to predict if the picture shows a dog or not, but it has never seen a weird-looking breed like Bergamasco Sheepdog during the training, what do you think if it tries to make predictions on a picture of Bergamasco Sheepdog as follows:

The machine learning model might make wrong predictions, because it looks very different from the dogs it was trained on.

Consider a machine learning model trained on the dataset below:



Rank the generalization gaps of the following three test datasets in descending order, and briefly explain the reasoning behind the ranking:

a. 

b. 

c. 

[Student answers the ranks in descending order:
c, b, a
]

Part 2: Machine Learning Classification

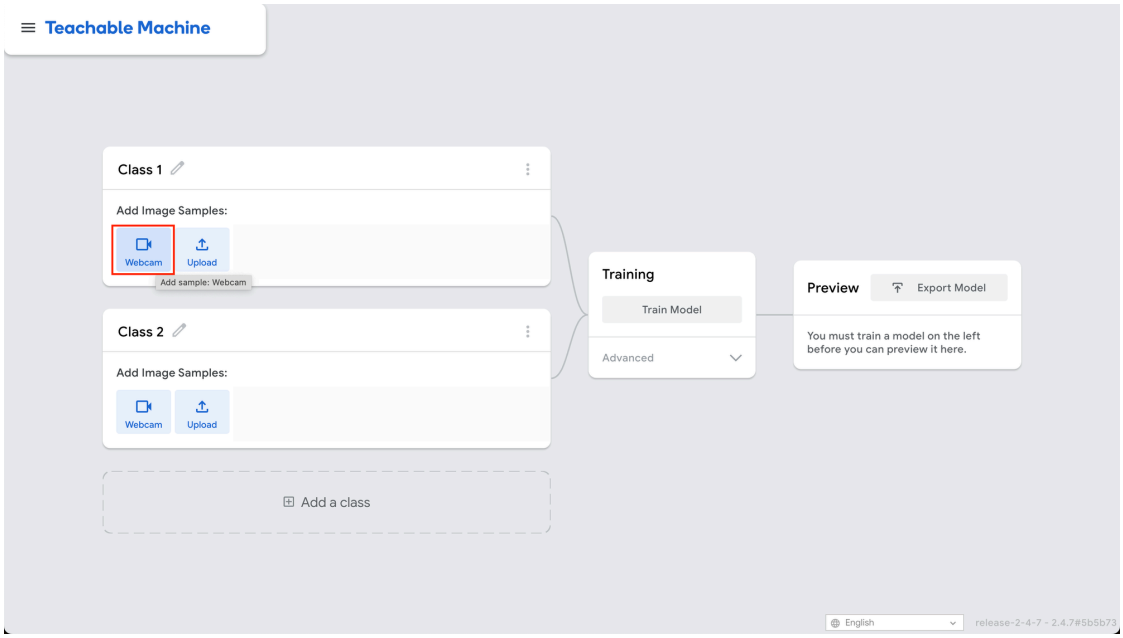
In this part, we will work on a hands-on machine learning project for classification using Teachable Machine, a web-based platform for training machine learning models without coding.

Prepare the training data

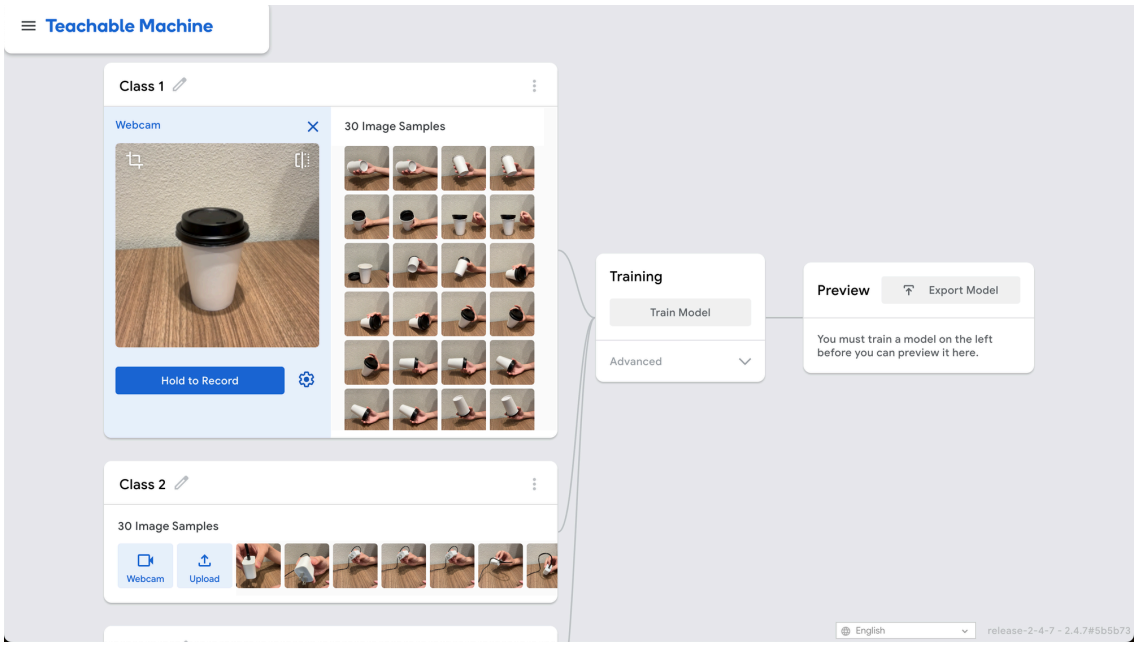
Let's begin by following the steps to prepare the training data.

1. Find three objects you have nearby. They could be pens, backpacks, notebooks, or whatever that can fit into the camera. (Try not to use paper cups. It is already used as an example in this assignment.)

2. Open the Teachable Machine, and click the "Webcam" button for "Class 1" to activate the webcam.



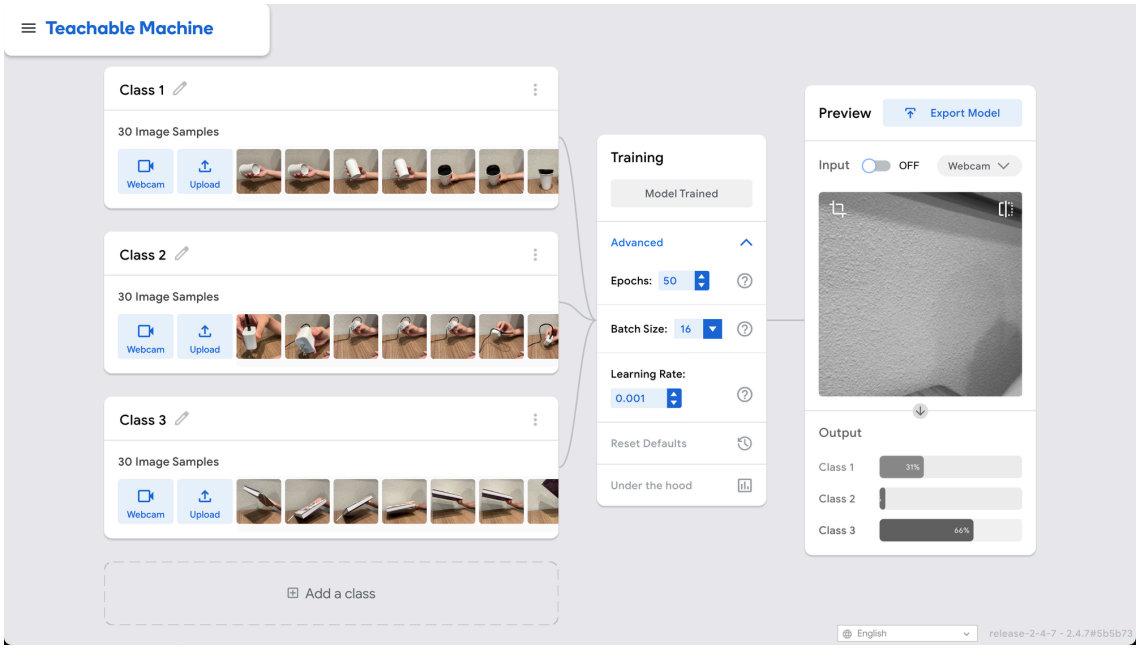
3. Hold the first object in the camera. Click the “Hold to Record” button to record images as the training data for “Class 1” until you have 30 images showing on the right. During recording, try to change the angle, position, and distance of the object so that the images have more variance.



4. Repeat steps 2 and 3 to create the training data for “Class 2” and “Class 3” with the second and third objects.

5. Click the “Train Model” button on the right and wait until the training is finished.

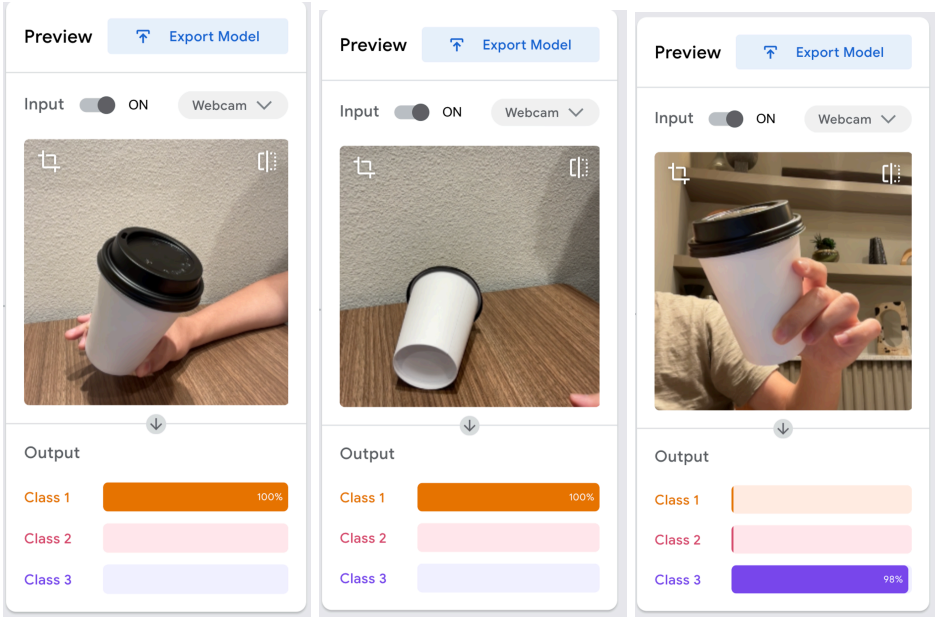
Upload a screenshot of the website showing the three classes and the model being trained. The screenshot should look similar to this:



[Student upload a screenshot]

Now, we can test our model with the “Preview” on the right. Put the object in the camera again, and the probabilities of the objects belonging to three classes will be calculated simultaneously at the bottom.

For each class, upload three screenshots of test data with **two successful tests** showing greater than 90% probability predicting the correct label and **one failed test** showing less than 50% probability predicting the correct label. The screenshots should look similar to this:



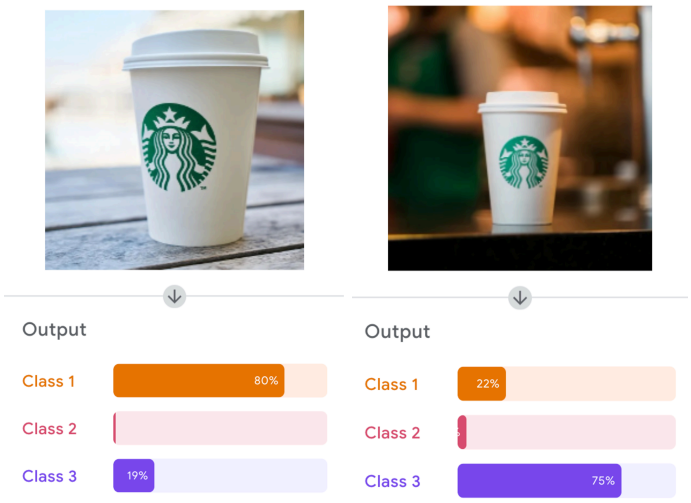
Class 1:
[Students upload 3 screenshots]

Class 2:
[Students upload 3 screenshots]

Class 3:
[Students upload 3 screenshots]

Generalization gap

Pick one of the classes and find 2 images of the same or similar object as the one you used to create the dataset from any public resources, e.g., Google image. Upload these images as the test data and record the predictions. Below are two examples of the test images from public resources. The first test still predicts it as Class 1 (cup), but with lower probability. The second test predicted it as Class 3 (book). This might be because the model has never seen cups with a logo or different background in its training dataset. This opens the generalization gap.



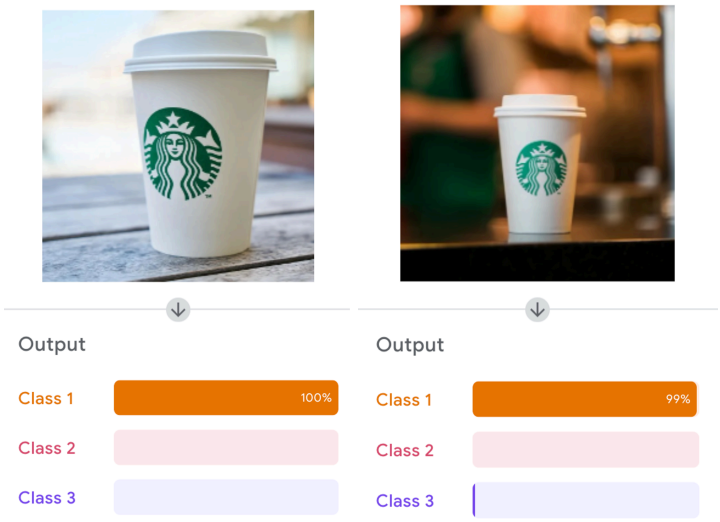
Upload the screenshots of your prediction results on the two test images:

[Students showing two test samples from internet:
Similar to the examples.
]

Does the model confidently predict the correct label? If not, write a few sentences explaining the reason:
[Students write a short essay:
No, the first test still predicts it as a cup, but with lower probability. The second test predicted it as the book. Because the model has never seen cups with a logo or different background. This opens the generalization gap.
]

How can we close the generalization gap? We add more training data similar to the test data! Now, find more images from public resources that look similar but not exactly the same as the two test images. Add these images into the training data and train another model. Does this model perform better on the two test data you found from the public resource? Upload the screenshots showing the improved performance of the new model on two test images.

[Students upload another two tests showing improved performance.



]

Evaluate machine learning models

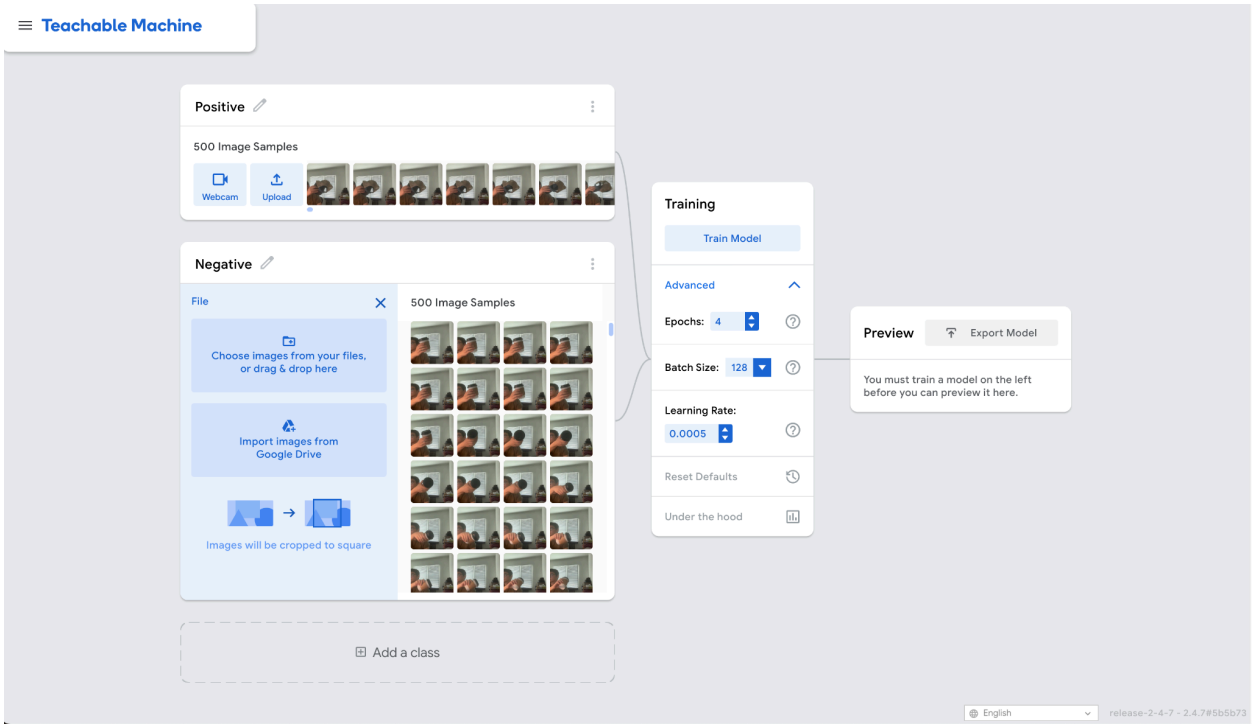
Now let's try to create a larger dataset. Follow the instructions:

1. Open the Teachable Machine again and start a new workspace. Name the two classes "Positive" and "Negative" respectively.

2. Choose one object and create 500 training data for "Positive." Use "Hold to Record" to record fast. Make sure you keep rotating and moving the object to create a diverse training dataset.

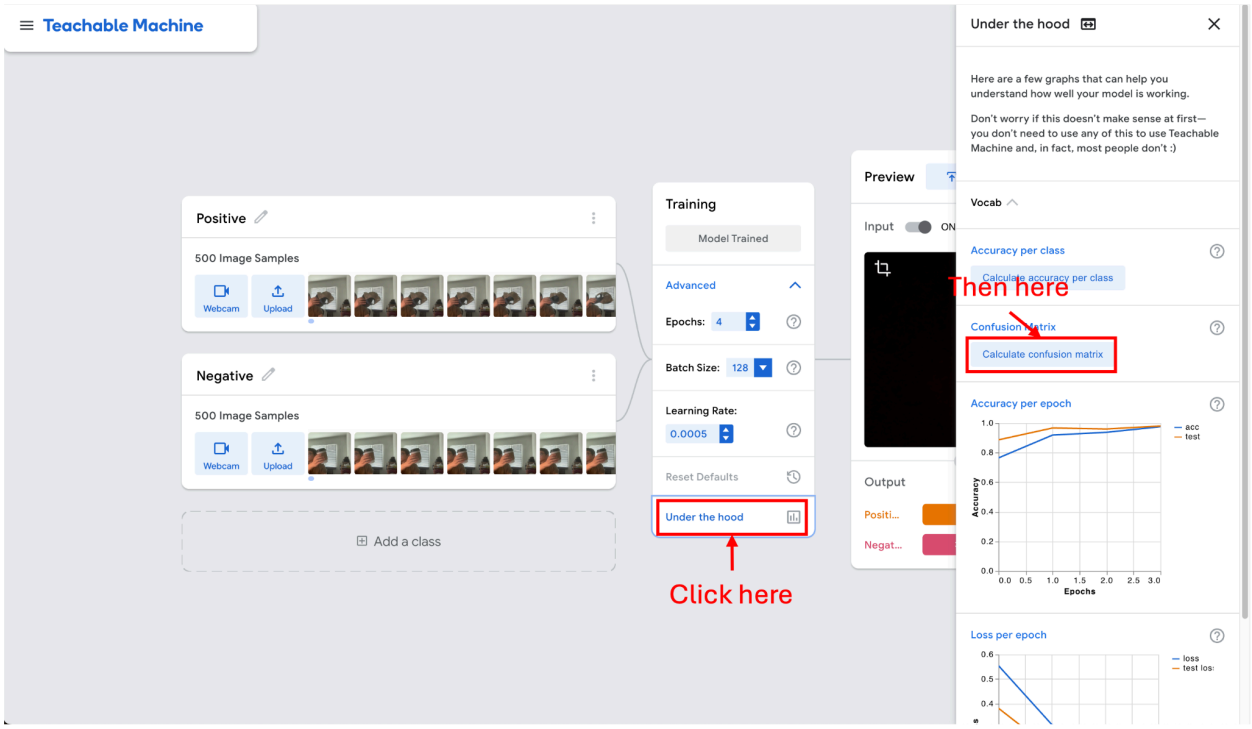
3. Choose another object and create 500 training data for "Negative." Use "Hold to Record" to record fast. Make sure you keep rotating and moving the object to create a diverse training dataset.

4. Click "Advanced" and set the training parameters as follows: Epochs: 4, Batch Size: 128, Learning Rate: 0.0005. Once finished, your workspace should look like this:

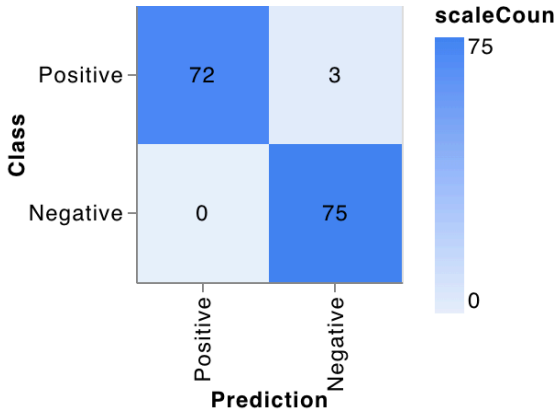


5. Click “Train Model” and wait until the training finishes.

6. Click “Under the Hood”, then click “Calculate confusion matrix” on the right panel.



7. Trainable machine will test the model on 30% of the training data and report the confusion matrix here. A confusion matrix will show up similar to the figure below:



- The top-left number stands for the number of images that belong to “Positive” and the model correctly predicts it as “Positive.” This number is also called “True Positive” or TP.
- The bottom-left number stands for the number of images that belong to “Negative” and the model falsely predicts it as “Positive.” This number is also called “False Positive” or FP.
- The top-right number stands for the number of images that belong to “Positive” and the model falsely predicts it as “Negative.” This number is also called “False Negative” or FN.
- The bottom-right number stands for the number of images that belong to “Negative” and the model correctly predicts it as “Negative.” This number is also called “True Negative” or FN.

Upload a picture of your confusion matrix:
[Students upload their confusion matrix]

Accuracy is the metric that reflects the performance of the model on the entire test set. It is the sum of all true predictions divided by the number of test data, or
$$
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}
$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Calculate your **Accuracy**:
[Students enter a number in the blank]

However, **accuracy** does not reflect performance of the model predicting each class. The model might achieve high **accuracy**, but completely fail to predict one of the classes. We will see examples soon. To mitigate this, another two metrics have been developed: **recall** and **precision**.

Precision is the metric that reflects the performance of the model predicting “Positive” data. It is the number true predictions of “Positive” divided by the sum of both true and false predictions of “Positive,” or
$$
\text{Precision} = \frac{TP}{TP + FP}
$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

Calculate your **precision**:
[Students enter a number in the blank]

Recall is the metric that reflects the performance of the model predicting “Negative” data. It is the number true predictions of “Negative” divided by the sum of both true and false predictions of “Negative,” or
$$
\text{Recall} = \frac{TN}{TN + FN}
$$

$$\text{Recall} = \frac{TN}{TN + FN}$$

Calculate your **recall**:
[Students enter a number in the blank]

Now, delete the images under class “Negative” until it has 250 images. To save time, you may also download all the images, delete the “Negative” class, create a new empty “Negative” class, and upload only 250 images of the original “Negative” training data.

Train the model again and compute the confusion matrix. Upload a picture of your confusion matrix:
[Students upload their confusion matrix]

Calculate the **accuracy**, **precision**, and **recall**:

[Students fill out the blanks

Accuracy: ___
Precision: ___
Recall: ___

]

Now, continue to delete the images under class “Negative” until it has 100 images. Train the model again and compute the confusion matrix. Upload a picture of your confusion matrix:
[Students upload their confusion matrix]

Calculate the **accuracy**, **precision**, and **recall**:
[Students fill out the blanks

Accuracy: ___
Precision: ___
Recall: ___

]

Does the model still perform well at predicting the “Negative?” Support your answer with the **accuracy**, **precision**, and **recall** metrics. Briefly explain how the change of the training data affects the performance of the model.
[Student write a short essay as follows:

]