

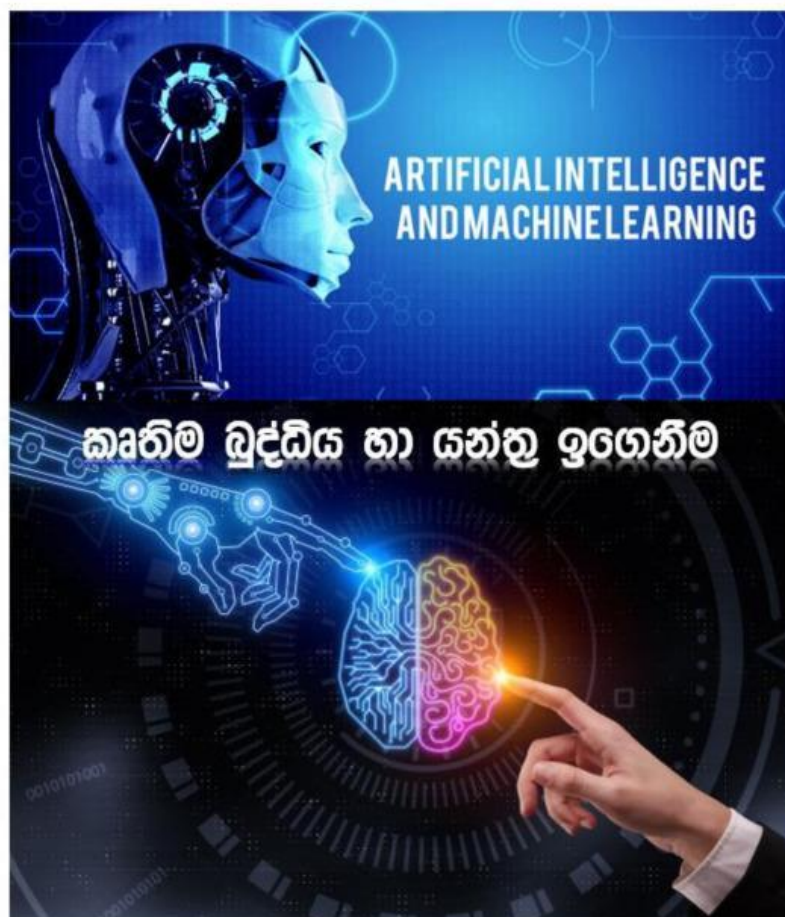
Project 180



Coding School



AI and Machine Learning

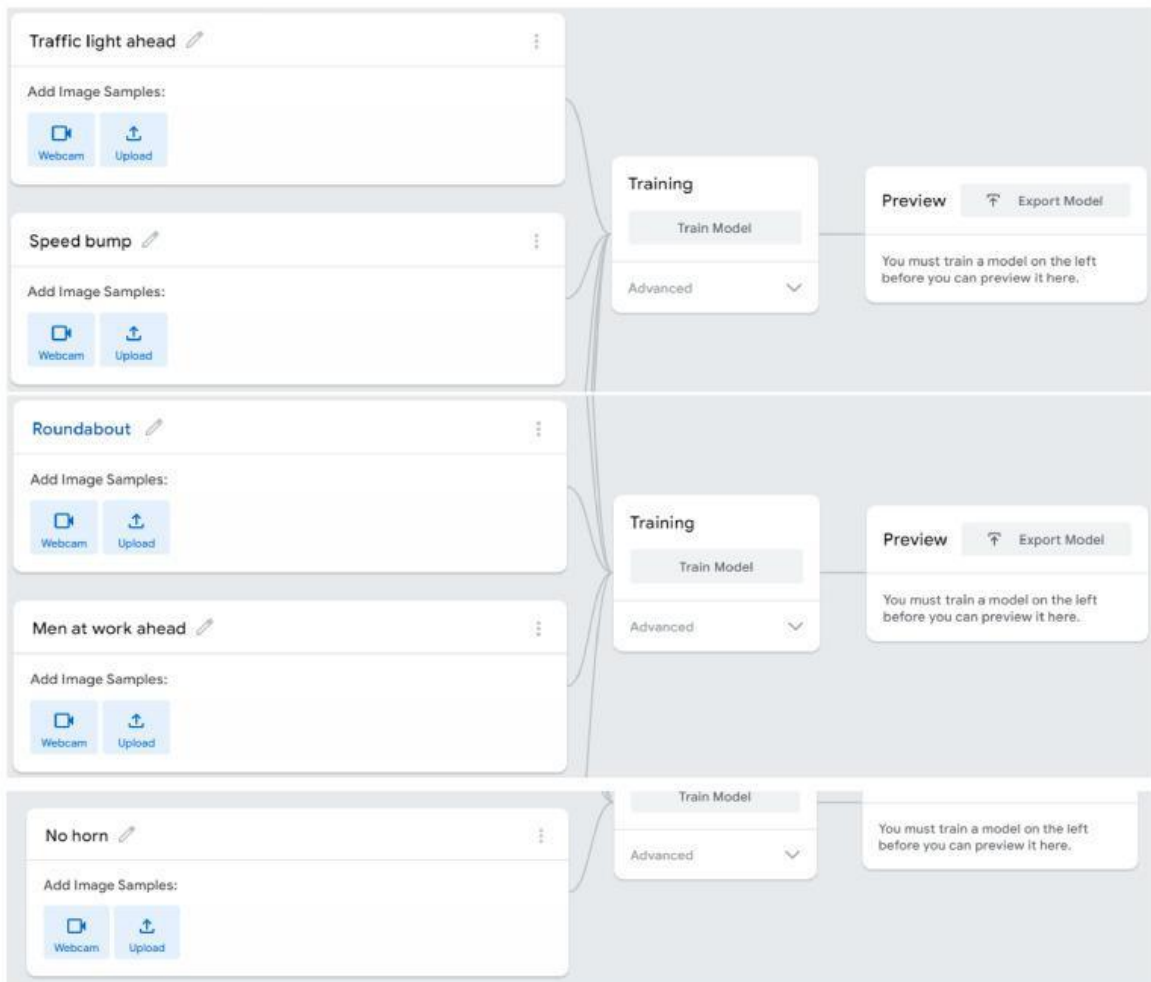


[See the web page](#)

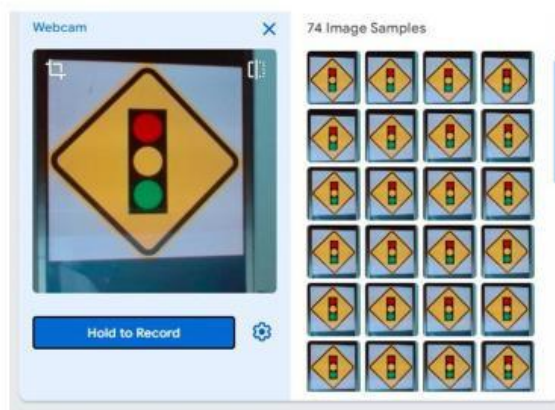
[Start here](#)

Let's see how to correctly perform image classification using Teachable Machine. Let's create a traffic sign recognition web page.

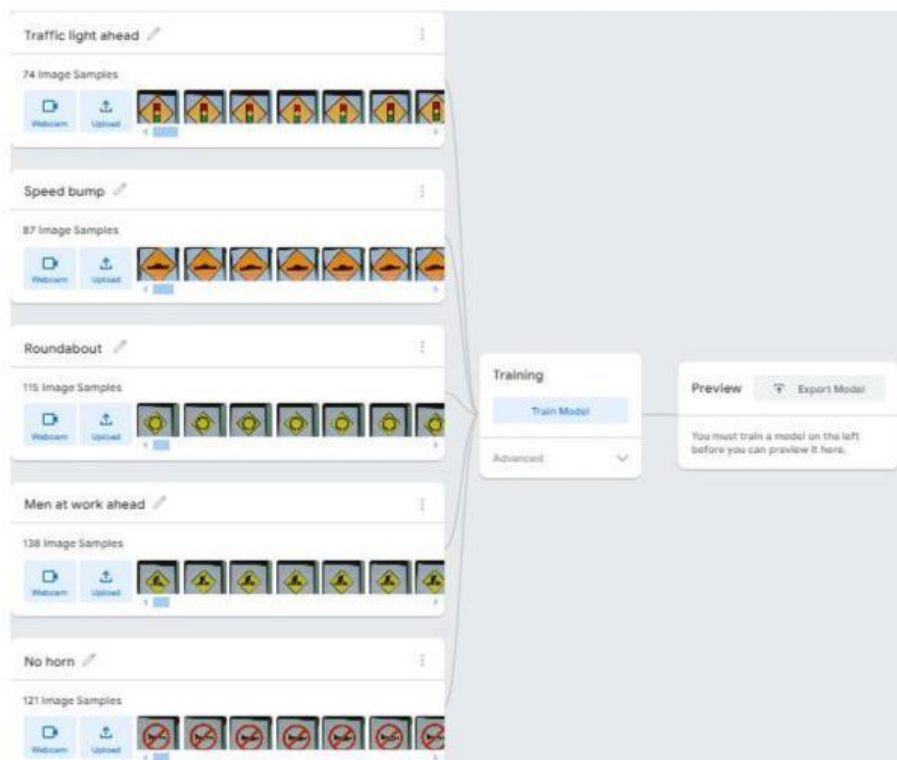
- ❖ First, <https://teachablemachine.withgoogle.com> go to the website and start training by getting started. Go into the Image project and start training the model.
- ❖ Add the class in the model as follows.



- ❖ Then provide samples for each class using the web cam.



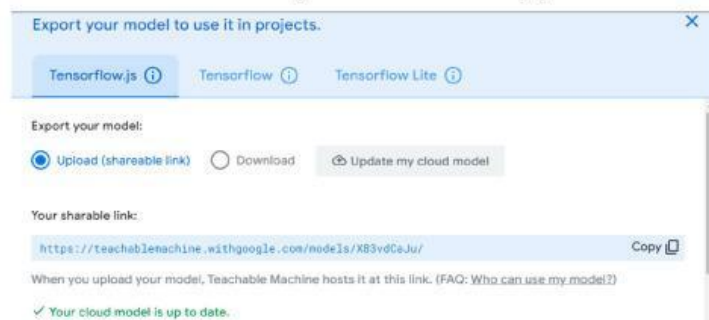
- ❖ Provide Samples for all Classes prepared as above.



- ❖ Then train the model.
- ❖ After training the model, you can see the prediction as below from the preview.



- ❖ Now upload the model to export it. Then copy the sharable link.



- ❖ Then let's start creating the website.
- ❖ Libraries are added as follows in the project you are given to start with.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Teachable Machine Image Model</title>
  <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@latest/dist/tf.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/@teachablemachine/image@latest/dist/teachablemachine-image.min.js"></script>
  <link rel="stylesheet" href="style.css">
</head>
```

- ❖ Set the dev tag in the body tag as follows.

```
<body>
  <div id="webcam-container"></div>
  <div id="label-container"></div>
  <div id="result-container"></div>
  <script src="script.js"></script>
</body>
```

- ❖ After that import the js file.
- ❖ Adjust the css as follows in the Style.css file.

```
1 #webcam-container,
2 #label-container,
3 #result-container {
4   margin-top: 20px;
5 }
6 #webcam-container {
7   display: flex;
8   justify-content: center;
9 }
10 #label-container div {
11   font-size: 20px;
12   margin: 5px;
13 }
14 #result-container {
15   font-size: 24px;
16   font-weight: bold;
17   text-align: center;
18 }
--
```

- ❖ Prepare the javascript code in the Script.js file as follows.
- ❖ First, add a variable to store the url of the model created and trained above.

```
const URL = "https://teachablemachine.withgoogle.com/models/BbkmB9R_C/";
```

For the url here, give the sharable link you copied above.

- ❖ Then create some more variables as follows.

```
let model, webcam, labelContainer, resultContainer, maxPredictions;
```

- ❖ Create a function as init as follows.

```
async function init() {  
  const modelURL = URL + "model.json";  
  const metadataURL = URL + "metadata.json";  
  
  model = await tmImage.load(modelURL, metadataURL);  
  maxPredictions = model.getTotalClasses();  
  
  const flip = true;  
  webcam = new tmImage.Webcam(200, 200, flip);  
  await webcam.setup();  
  await webcam.play();  
  window.requestAnimationFrame(loop);  
  
  document.getElementById("webcam-container").appendChild(webcam.canvas);  
  labelContainer = document.getElementById("label-container");  
  resultContainer = document.getElementById("result-container");  
  for (let i = 0; i < maxPredictions; i++) {  
    labelContainer.appendChild(document.createElement("div"));  
  }  
}
```

- ❖ After creating it, call the function below it.

```
init();
```

- ❖ Then create a function as a loop as follows.

```
async function loop() {  
  webcam.update();  
  await predict();  
  window.requestAnimationFrame(loop);  
}
```

- ❖ Finally, create the predict function as follows.

```
async function predict() {  
  const prediction = await model.predict(webcam.canvas);  
  let highestProbability = 0;  
  let highestClass = "";  
  for (let i = 0; i < maxPredictions; i++) {  
    const classPrediction =  
      prediction[i].className + ": " + prediction[i].probability.toFixed(2);  
  
    labelContainer.childNodes[i].innerHTML = classPrediction;  
    if (prediction[i].probability > highestProbability) {  
      highestProbability = prediction[i].probability;  
      highestClass = prediction[i].className;  
    }  
  }  
  
  resultContainer.innerHTML = "Detected Sign: " + highestClass;  
}
```

- ❖ You can display the result as text or as an image in the result container.
- ❖ After setting it up as above, the web cam will open and prediction will start.



Traffic light ahead: 0.13
Speed bump: 0.11
Roundabout : 0.27
Men at work ahead: 0.15
No horn: 0.33

Detected Sign: No horn

- ❖ Now let's present each road sign to the web cam and check how to detect it correctly.



Traffic light ahead: 1.00
Speed bump: 0.00
Roundabout : 0.00
Men at work ahead: 0.00
No horn: 0.00

Detected Sign: Traffic light ahead



Traffic light ahead: 0.00
Speed bump: 0.00
Roundabout : 1.00
Men at work ahead: 0.00
No horn: 0.00

Detected Sign: Roundabout