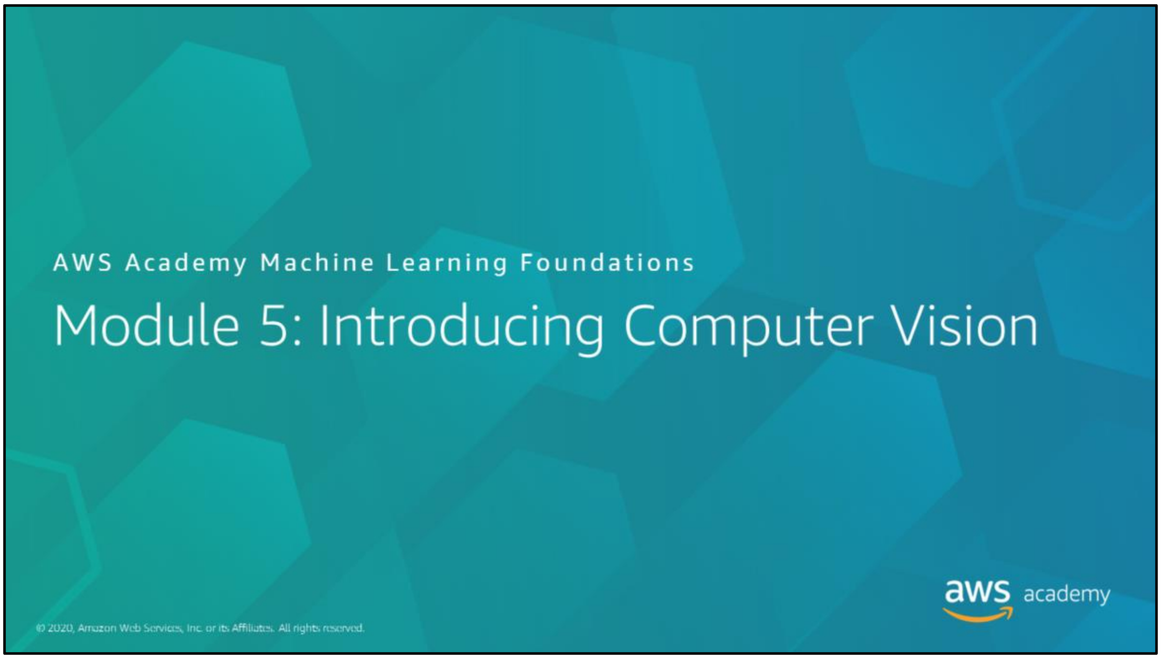
AWS Academy Machine Learning Foundations
Module 05 Student Guide
Version 1.0.3

200-ACMLFO-10-EN-SG

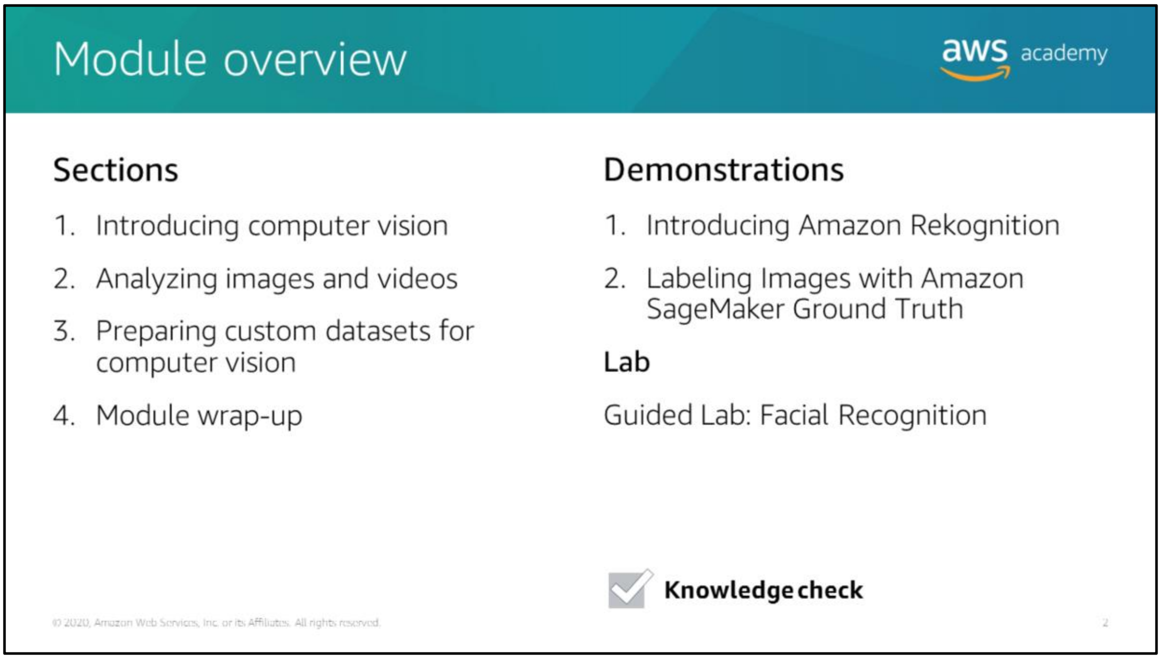
AWS Academy Machine Learning Foundations

Contents

Module 5: Introducing Computer Vision 4



Welcome to Module 5: Introduction to Computer Vision.

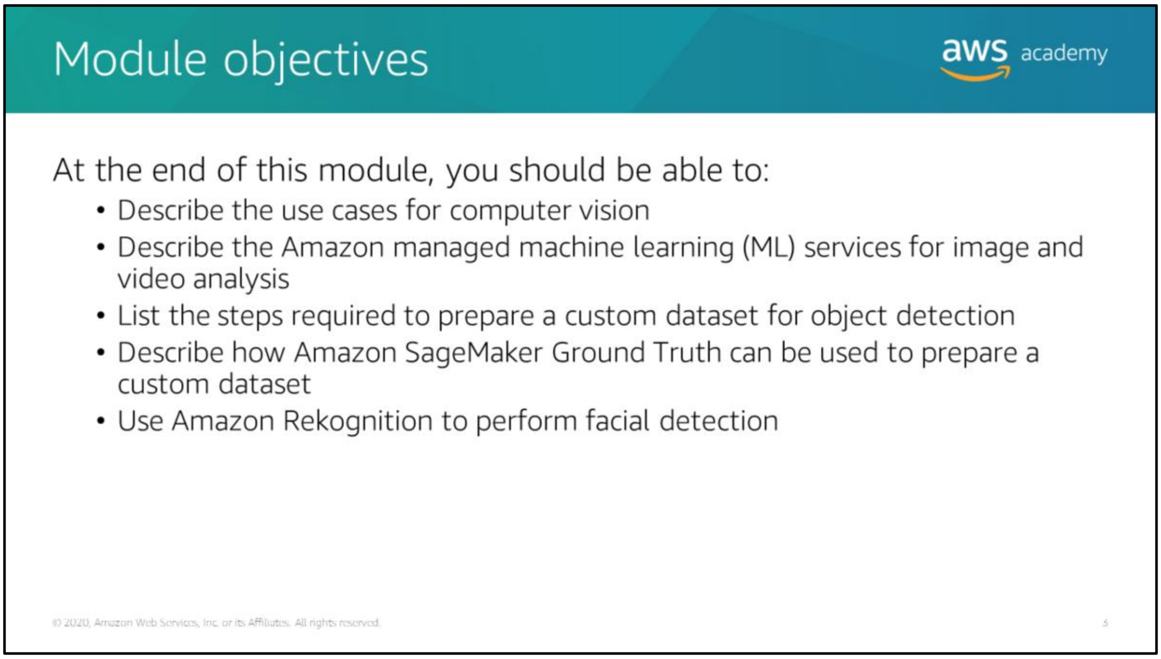


This module will address the following sections:

1. Introduction to computer vision
2. Image and video analysis
3. Preparing custom datasets for object detection
4. Guided Lab: Facial Recognition
5. Module wrap-up

The module includes educator-led demonstrations that show you how Amazon Rekognition can evaluate images and video and how to use Amazon SageMaker Ground Truth.

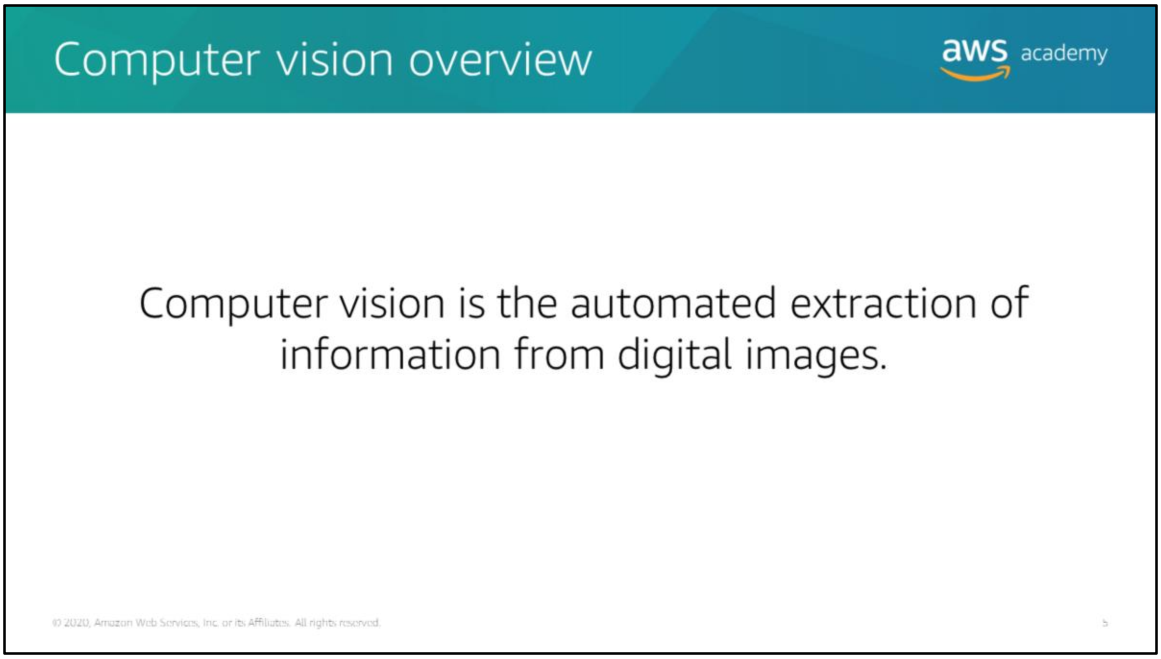
Finally, you will be asked to complete a knowledge check that will test your understanding of key concepts that are covered in this module.



At the end of this module, you should be able to:

- Describe the use cases for computer vision
- Describe the Amazon managed ML services available for image and video analysis
- List the steps required to prepare a custom dataset for object detection
- Describe how Amazon SageMaker Ground Truth can be used to prepare a custom dataset
- Use Amazon Rekognition to perform facial detection

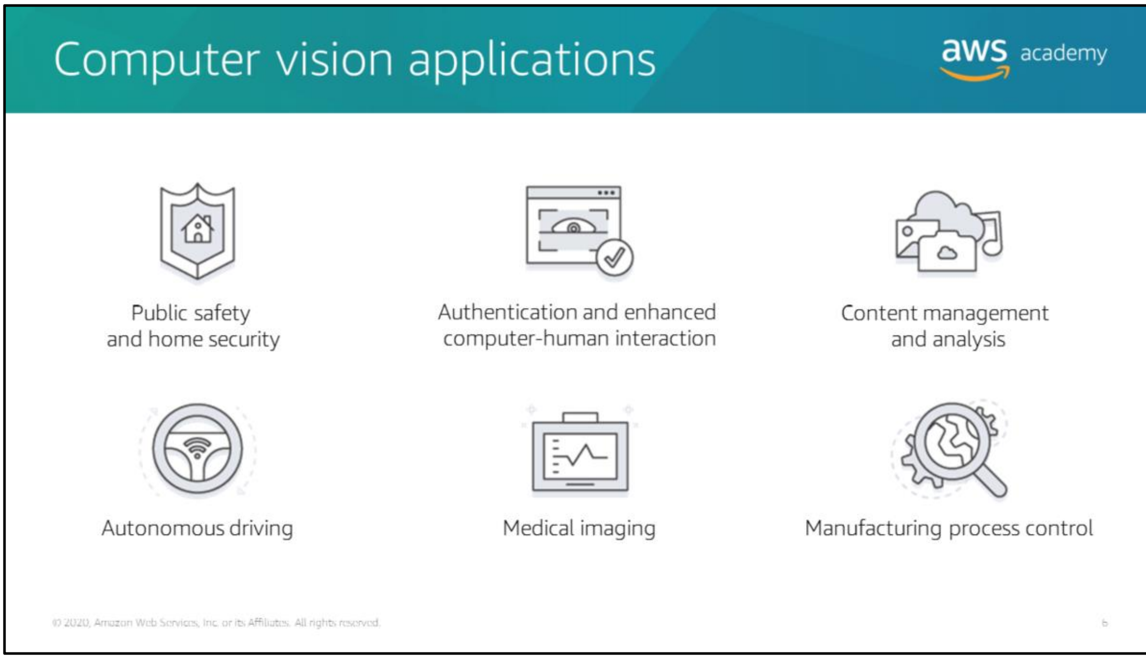
Introducing Section 1: Introduction to computer vision.



Computer vision is an exciting space in machine learning. The advances in computing power and algorithms over the last 10 years have led to an increase in capabilities and easier access to computer vision technologies.

You can think of computer vision as the automated extraction of information from digital images.

Computer vision enables machines to identify people, places, and things in images with accuracy at or above human levels, with greater speed and efficiency. Often built with deep learning models, computer vision automates the extraction, analysis, classification, and understanding of useful information from a single image or a sequence of images. The image data can take many forms, such as single images, video sequences, views from multiple cameras, or three-dimensional data.



Some of the primary use cases for computer vision include these examples.

Public safety and home security
Computer vision with image and facial recognition can help to quickly identify unlawful entries or persons of interest. This process can result in safer communities and a more effective way of deterring crimes.

Authentication and enhanced computer-human interaction
Enhanced human-computer interaction can improve customer satisfaction. Examples include products that are based on customer sentiment analysis in retail outlets or faster banking services with quick authentication that is based on customer identity and preferences.

Content management and analysis
Millions of images are added every day to media and social channels. The use of computer vision technologies—such as metadata extraction and image classification—can improve efficiency and revenue opportunities.

Autonomous driving
By using computer-vision technologies, auto manufacturers can provide improved and safer self-driving car navigation, which can help realize autonomous driving and make it a reliable transportation option.

Medical imaging
Medical image analysis with computer vision can improve the accuracy and speed of a patient's medical diagnosis, which can result in better treatment outcomes and life expectancy.

Manufacturing process control
Well-trained computer vision that is incorporated into robotics can improve quality assurance and operational efficiencies in manufacturing applications. This process can result in more reliable and cost-effective products.

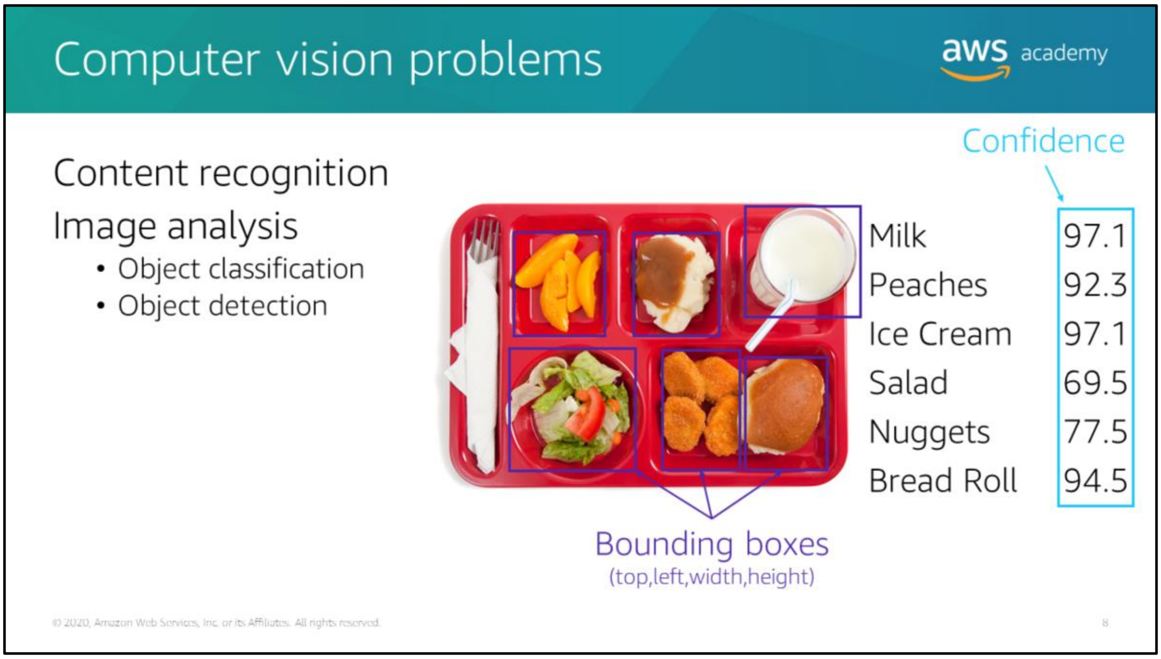
Classification in machine learning is used to decide which category or categories that a picture or object belongs to. This process is no different than any other classification problem for machine learning.

In the picture here, what is represented? Is it breakfast, lunch, or dinner? Would the classification only be food?

The answer depends on the model that you use to perform the classification. Models must be trained, and the training data provides the algorithm with the data to learn from.

Say that you have a model that was trained with pictures of different types of food. You might expect a result that the image outputs categories of *Milk*, *Cookie*, *Orange*, *Hamburger* (or is it beef burger, or a vegetarian burger?), *Salad*, and *French Fries*. If you trained the model with different images, you might have classified the objects in the image as *Tray*, *Cutlery*, and *Napkin*.

When you have multiple classes, this is known as a *multi-class classification problem*. When you have only two classes, this is known as a *binary classification problem*.

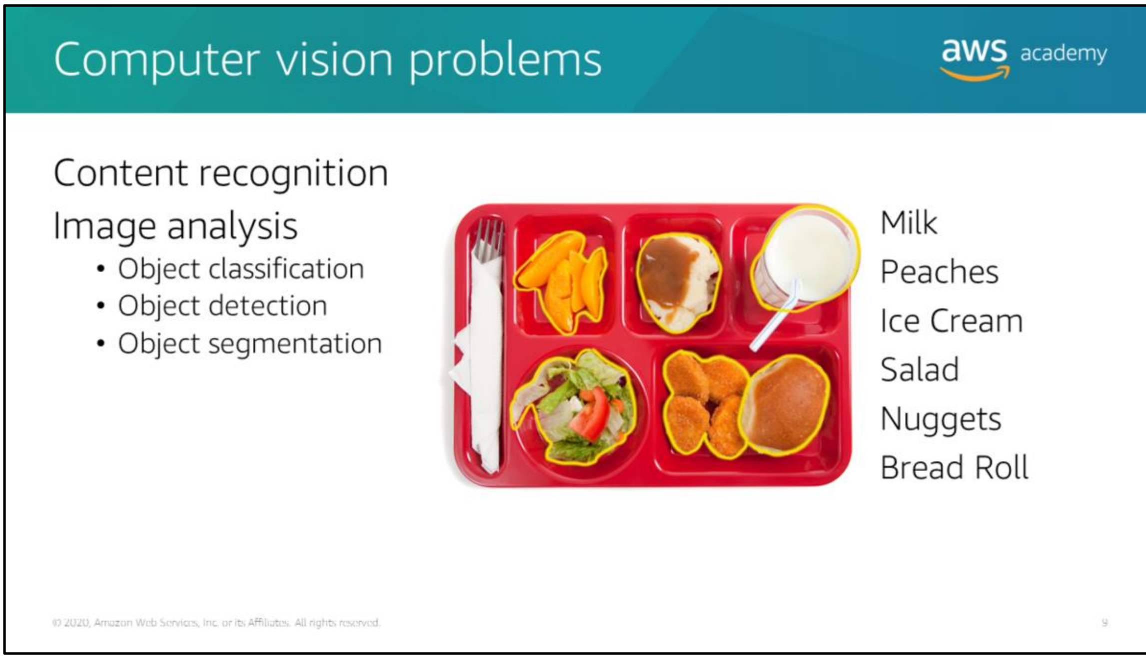


When you work with images, you might want to know both what kind of objects are in the image and the locations of those objects.

Object detection provides the categories of the image and the location of the objects in the image. The location is provided by a set of coordinates for a box that surrounds the image, which is known as the *bounding box*.

Bounding boxes for object detection typically provide top, left, width, and height coordinates that surround the images. You can use these coordinates in your applications.

When objects are detected in an image, a confidence number is usually associated with that object. This percentage indicates how probable it is that the object belongs to a specific class. This confidence level is important when you want to determine an action that is based on object detection, especially in applications that use facial detection.



At the time of writing, object segmentation (also known as semantic segmentation) is a key problem in the field of computer vision. Object segmentation is similar to object detection, but you go into more detail to get fine boundaries for each detected object. It is a fine-grained inference for predicting each pixel in the image.

Applications that require object segmentation include autonomous vehicles and advanced computer-human interactions. Object segmentation is not covered in this course.

Computer vision use cases

aws academy

Content recognition

Video analysis

- Instance tracking

Pathing – You can capture the path of people in the scene. For example, you can use the movement of athletes during a game to identify plays for post-game analysis.

With Amazon Rekognition Video, you can capture the position of each person in a video. The TrackPersons operation detects people and how they move, even when the camera is in motion. It can also attribute motion to the same person when their face is blocked or if they move in and out of the frame. The TrackPersons operation returns time segments and confidence scores.

Computer vision use cases

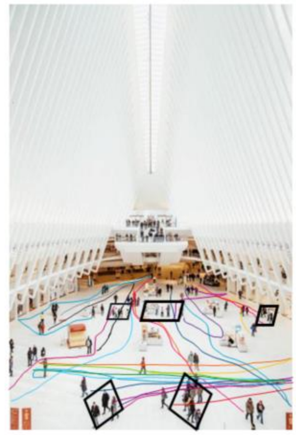
aws academy

Content recognition

Video analysis

- Instance tracking
- Action recognition

Analyze shopper behavior and density in your retail store by studying the path that each person follows

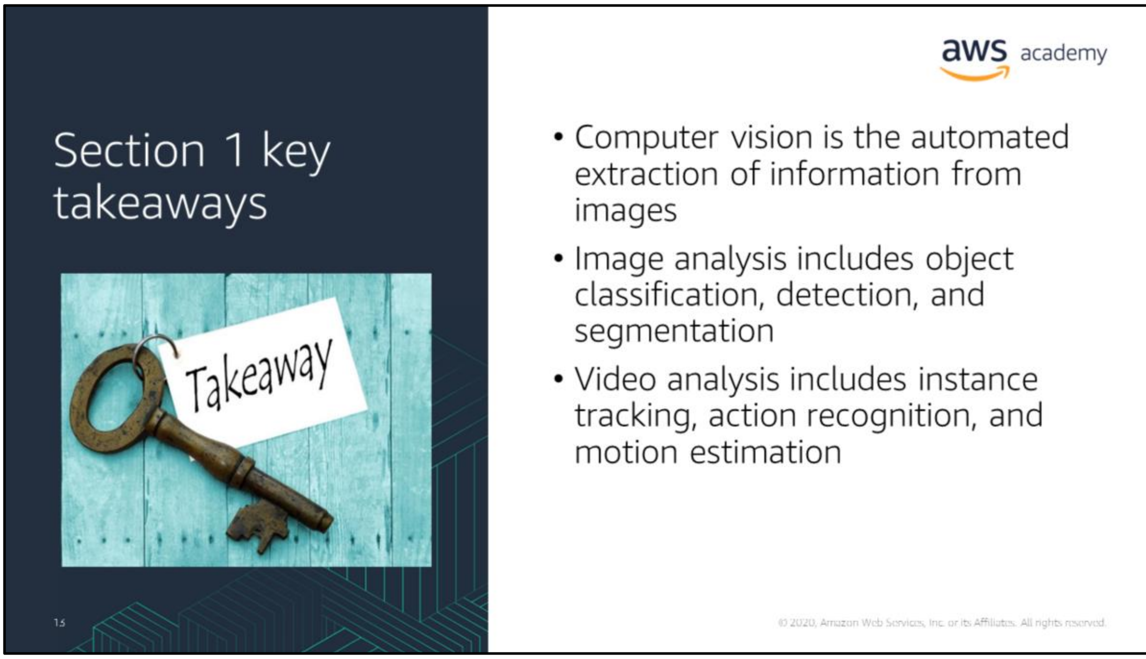


You can analyze shopper behavior and density in your retail store by studying the path that each person follows. By using face analysis, you can also understand the average age ranges, gender distribution, and emotions expressed by the shoppers without identifying them.

Amazon Rekognition Video enables you to automatically identify thousands of objects. Example objects might include vehicles or pets; scenes like a city, beach, or wedding; and activities like delivering a package or dancing. Amazon Rekognition Video relies on motion in the video to accurately identify complex activities, such as *blowing out a candle* or *extinguishing fire*.

Some examples from the image might include:

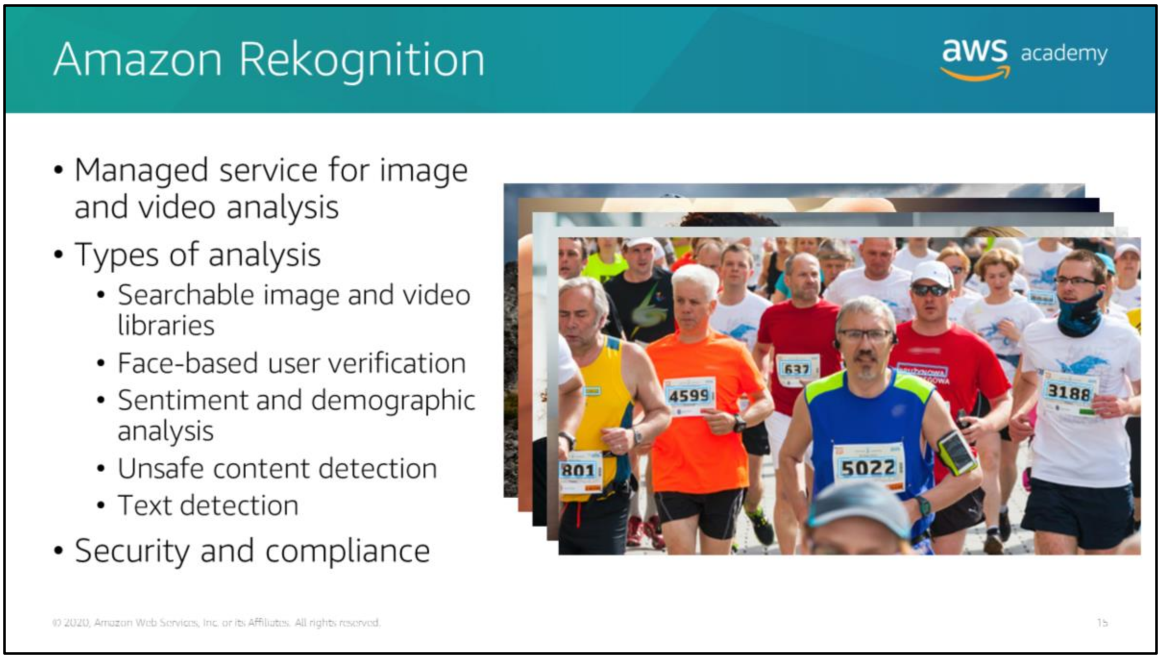
- Capturing the batter's accuracy, the pitcher's pitching style, the type of pitch (slow ball, slider, and others), the inning, and the batter's performance versus the specific pitcher. All that data could (and does!) get used by managers to coach players on how to improve their performance. Coaches can also use the data during the game to make game-time decisions.
- Initiating various actions based on the speed of the ball leaving the bat and its trajectory. A hit that is calculated by an ML model could lead to:
    1. An audio or visual warning about a possible foul ball into the crowd.
    2. A preemptive alarm that a hit has a high probability of being a home run. This would enable events that follow a homerun to be well-timed and automated (such as music or fireworks when the homerun is hit by the home team).



Some key takeaways from this section of the module include:

- *Computer vision* is the automated extraction of information from images
- From a practical point of view, you can divide computer vision into two distinct areas—image analysis and video analysis
  - Image analysis includes object classification, detection, and segmentation
  - Video analysis includes instance tracking, action recognition, and motion estimation

Introducing Section 2: Analyzing images and videos.

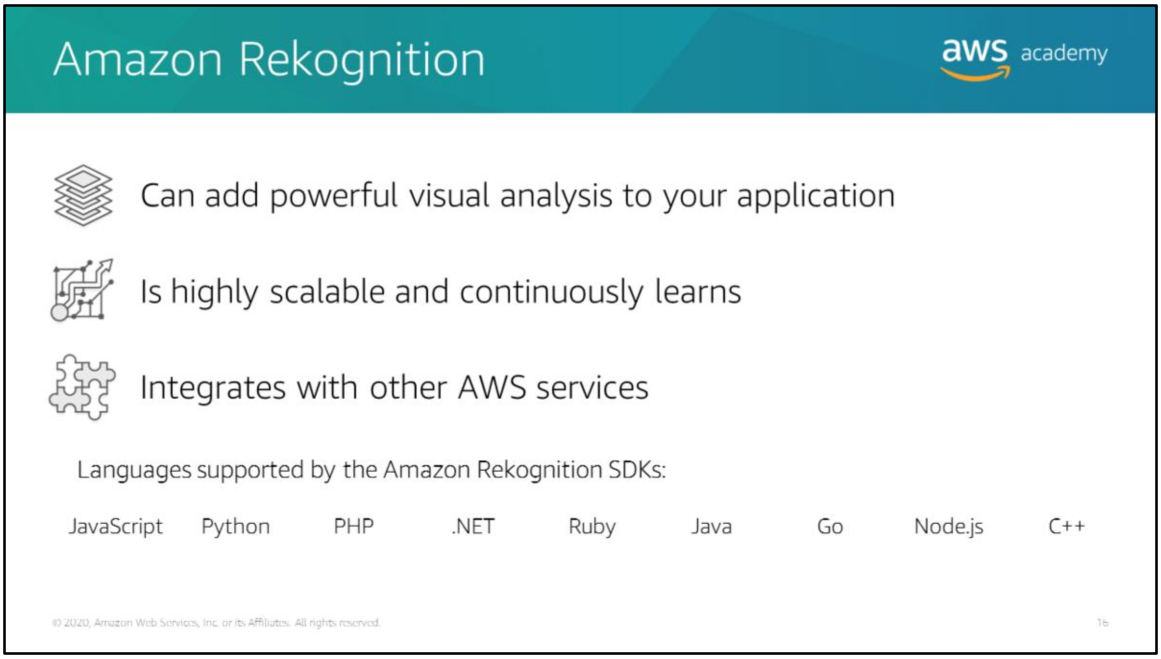


Amazon Rekognition is a computer vision service based on deep learning. You can use it to add image and video analysis to your applications.

Amazon Rekognition enables you to perform the following types of analysis:

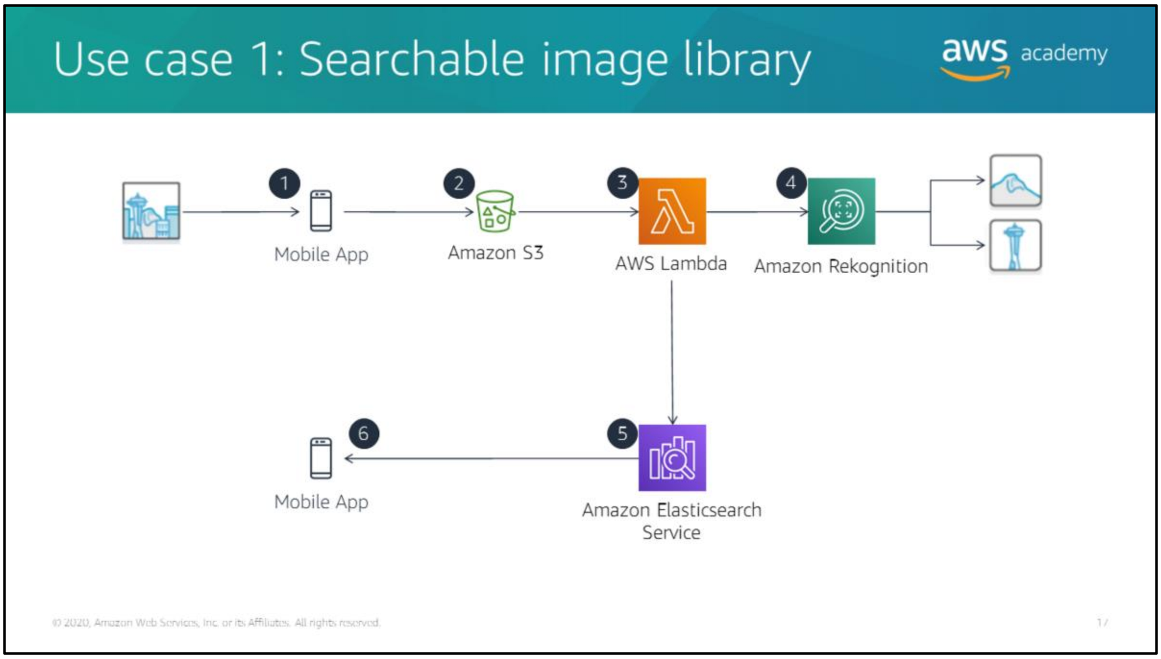
- **Searchable image and video libraries** – Amazon Rekognition makes images and stored videos searchable so that you can discover the objects and scenes that appear in them.
- **Face-based user verification** – Amazon Rekognition enables your applications to confirm user identities by comparing their live image with a reference image.
- **Sentiment and demographic analysis** – Amazon Rekognition interprets emotional expressions, such as happy, sad, or surprise. It can also interpret demographic information from facial images, such as gender.
- **Unsafe content detection** – Amazon Rekognition can detect inappropriate content in images and in stored videos
- **Text detection** – Amazon Rekognition Text in Image enables you to recognize and extract text content from images.

You need to check if the applications you build using Amazon Rekognition would fall under any regulatory restrictions as defined within your field or country. Security and compliance for Amazon Rekognition is a shared responsibility between AWS and the customer. For more information, see the AWS Compliance page (https://aws.amazon.com/compliance/)



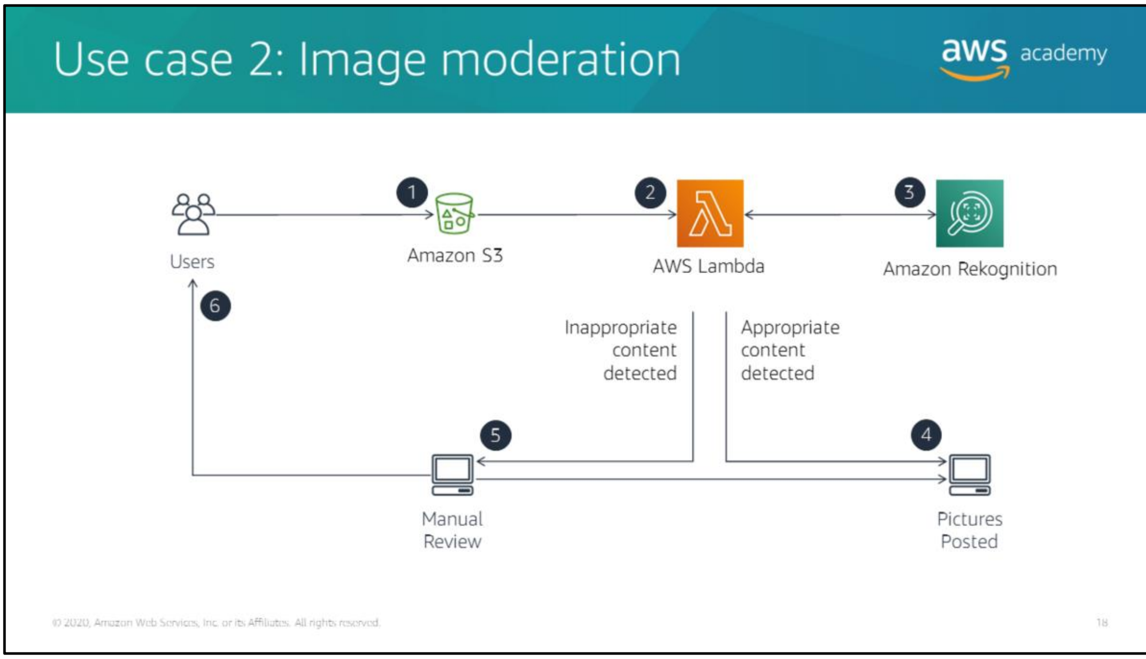
Amazon Rekognition is an AWS managed service that enables you to integrate image and video analysis into your applications. Because it’s a managed service, Amazon Rekognition hosts the machine learning models, maintains an API, and scales out to meet demand for you. You also benefit from a set of models that constantly learn and improve. It integrates with other AWS services, such as Amazon S3 for storage and AWS Identity and Access Management (IAM) for authentication and authorization. You only need to focus on building applications that use the API and, optionally, training the service to understand your unique business requirements.

Amazon Rekognition provides APIs, SDKs, and AWS Command Line Interface (AWS CLI) commands. You can use these resources to access and interact with Amazon Rekognition. The languages that the SDKs support include JavaScript, Python, PHP, .NET, Ruby, Java, GO, Node.js, and C++.



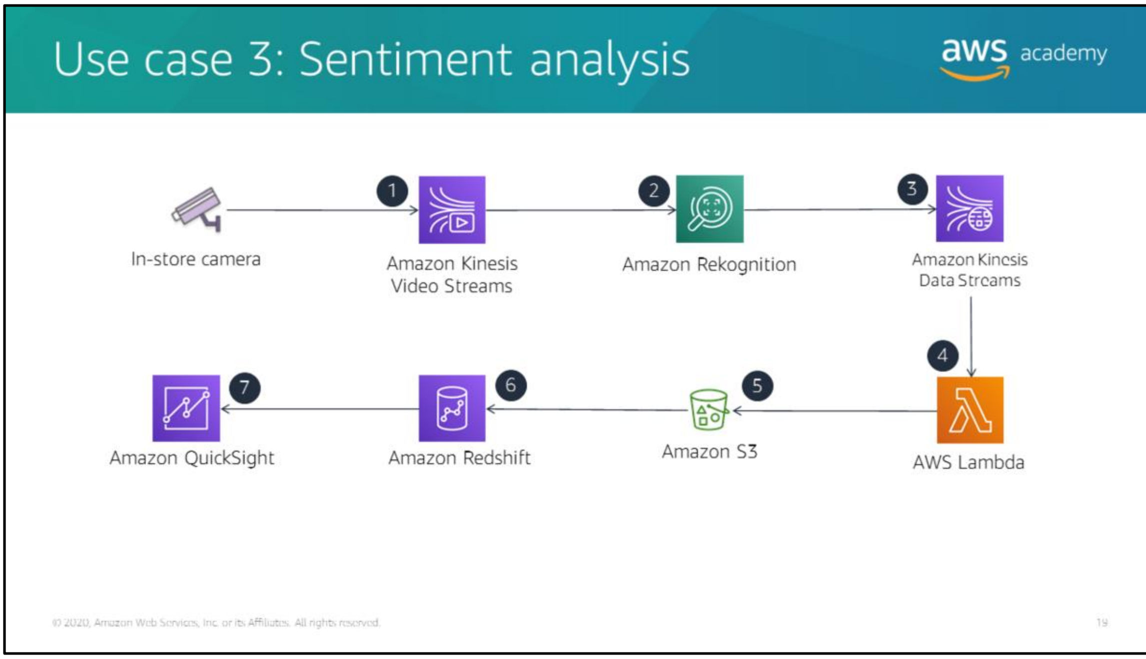
The diagram illustrates an image search capability where users can take pictures and receive information about the real estate properties that they are viewing.

1. The user takes a picture using their mobile device. The user then initiates a search, which causes the application to upload to Amazon S3.
2. Amazon S3 is configured to call other services when a write event occurs. In this case, the bucket passes the S3 path of the new object to AWS Lambda.
3. When the Lambda function is called, it uses the Amazon Rekognition SDK to call the service.
4. Amazon Rekognition analyzes the image, detects aspects of the property, creates labels, and passes the information back to Lambda as a JavaScript Object Notation (JSON) object.
5. Lambda then stores the labels and confidence score in Amazon Elastic Search.
6. Application users can now identify aspects of a property by using the objects that are detected in the image.



In this example architecture, the system checks uploaded images for inappropriate content. Like the previous example, processing begins when the user uploads content.

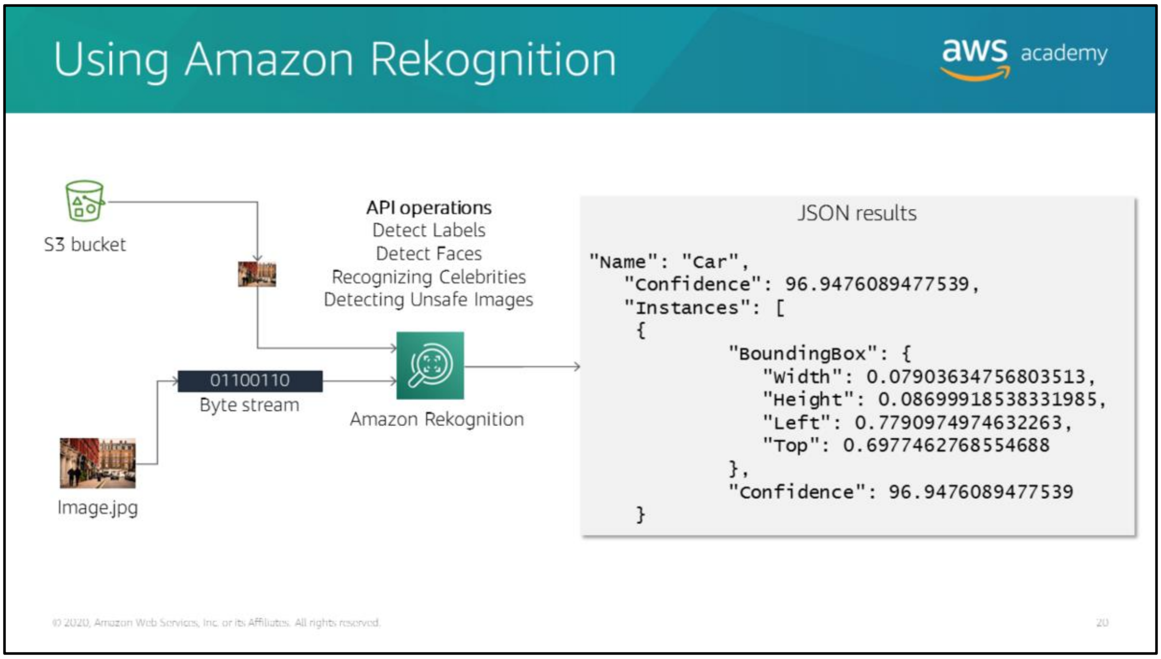
1. The user uploads an image to Amazon S3.
2. The S3 bucket is configured to call a Lambda function when an object is written to the bucket.
3. Lambda calls Amazon Rekognition via the SDK. Amazon Rekognition analyzes the images for inappropriate content and sends the response back to Lambda.
4. If the content is appropriate, the content is approved.
5. If the content is not appropriate, the content can be sent for manual inspection.
6. If not approved, notification is sent to the user.



In this final use case, the system analyzes a video feed for sentiment analysis.

.

1. An in-store camera captures video that is sent to a back office or a cloud-based application. Typically, an application like this will use Amazon Kinesis to stream the video.
2. The application uses the Amazon Rekognition SDK to send the video to Amazon Rekognition for further analysis. Visual sentiment is extracted, along with other attributes such as age, visual gender etc.
3. The discovered attributes are sent to Amazon Kinesis.
4. A Lambda function extracts the data from the stream.
5. Data is written to S3.
6. Periodically the data is loaded into Redshift.
7. Reports can be generated from the data in tools such as Amazon QuickSight.

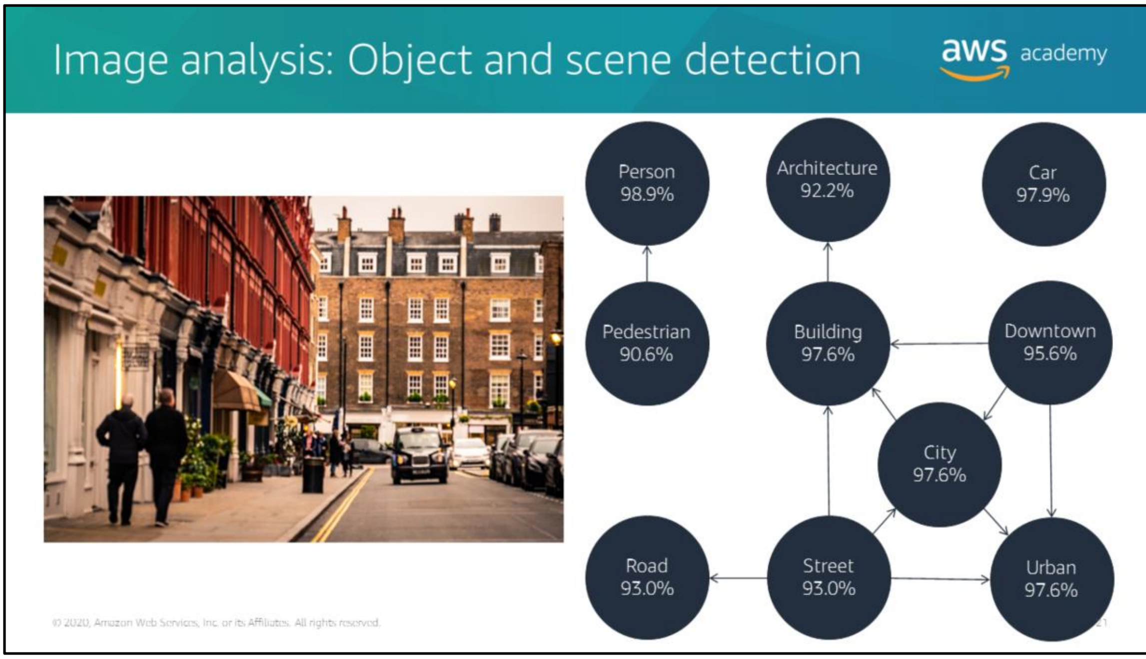


Amazon Rekognition is designed to integrate into your applications via the API and SDKs.

API operations are provided for detecting labels, faces, recognizing celebrities, and detecting unsafe images.

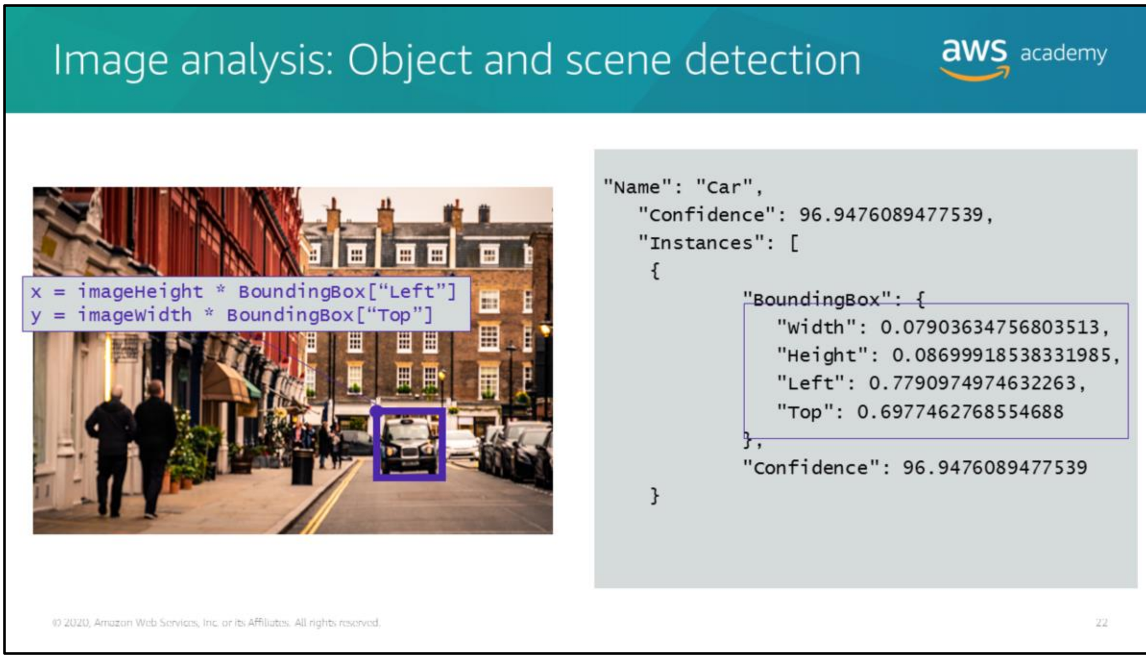
To perform a prediction, you must provide the service either an image object in Amazon S3, or upload a byte steam of an image. Images can be JPEG or PNG formats.

Amazon Rekognition processes the image, performs the prediction, and returns a JSON object with the results.



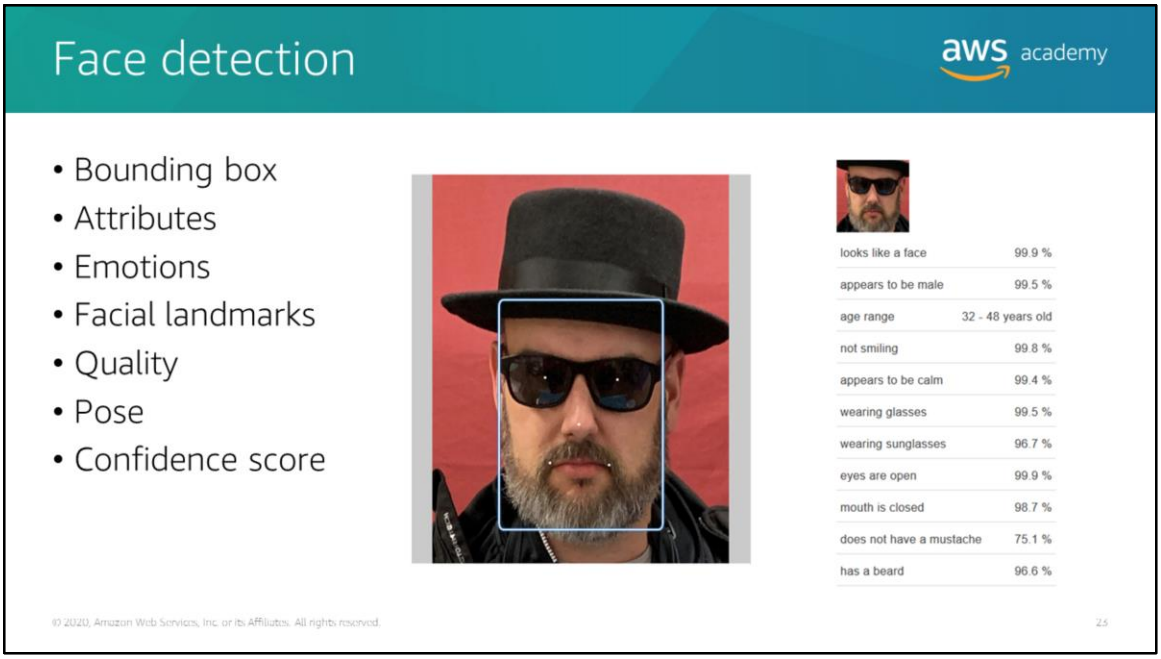
When Amazon Rekognition performs predictions, it often returns multiple labels. Each label has a confidence level. This confidence level indicates how likely the label was found in the image.

Like this example shows, labels can also have hierarchies.



When you find instances of objects, you must understand where in the image the detected object is. The results from Amazon Rekognition for each instance include a bounding box that contains the starting coordinate (*Top*, *Left*) and box dimensions (*Width*, *Height*). Like the example, you can use this information to determine the location of the detected object in the image.

It's important to note that all findings contain a confidence score. You can use the confidence score in your applications to tune your response to predictions. With a higher score, it's more likely that the object was correctly labeled.

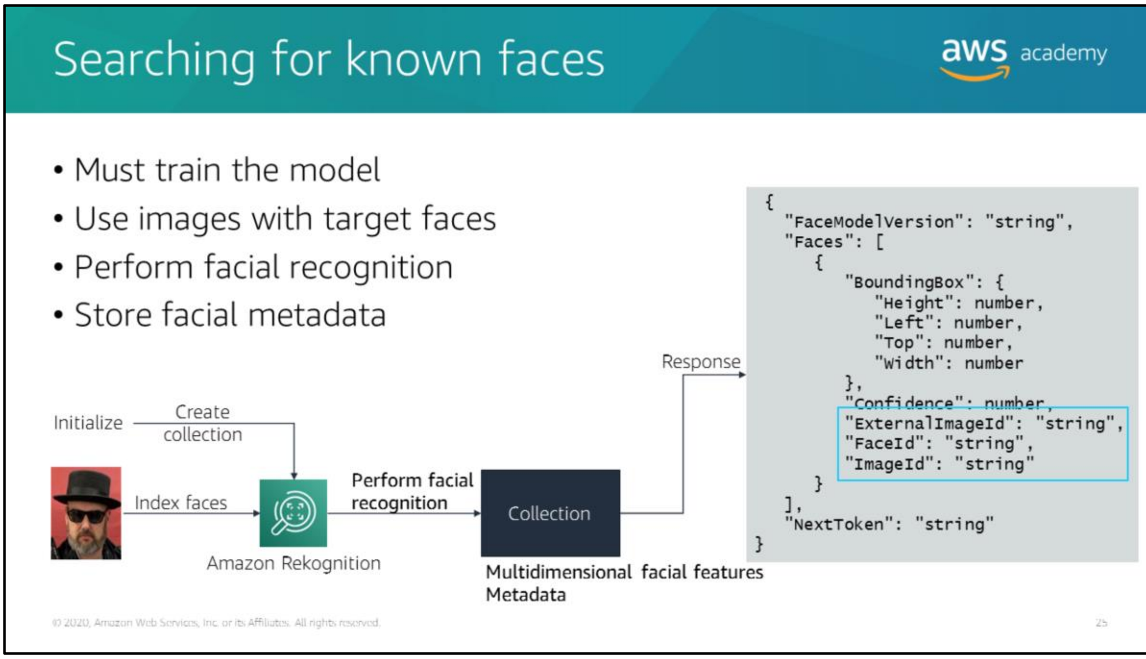


Facial detection uses a model that was tuned to perform predictions specifically for detecting faces and facial features, including:

- **Bounding box** – The coordinates of the bounding box that surrounds the face.
- **Confidence** – The level of confidence that the bounding box contains a face.
- **Facial landmarks** – An array of facial landmarks. For each landmark (such as the left eye, right eye, and mouth), the response provides the x and y coordinates.
- **Facial attributes** – A set of facial attributes, such as whether the face has a beard. For each attribute, the response provides a value. The value can be of different types, such as a Boolean type (whether a person is wearing sunglasses) or a string (whether the person is male or female). For most attributes, the response also provides a confidence score in the detected value for the attribute.
- **Quality** – Describes the brightness and the sharpness of the face.
- **Pose** – Describes the rotation of the face inside the image.
- **Emotions** – A set of emotions with confidence in the analysis.

Again, confidence is a feature here, and it is provided for each feature that was detected. The feature prediction is based only on visual observation.

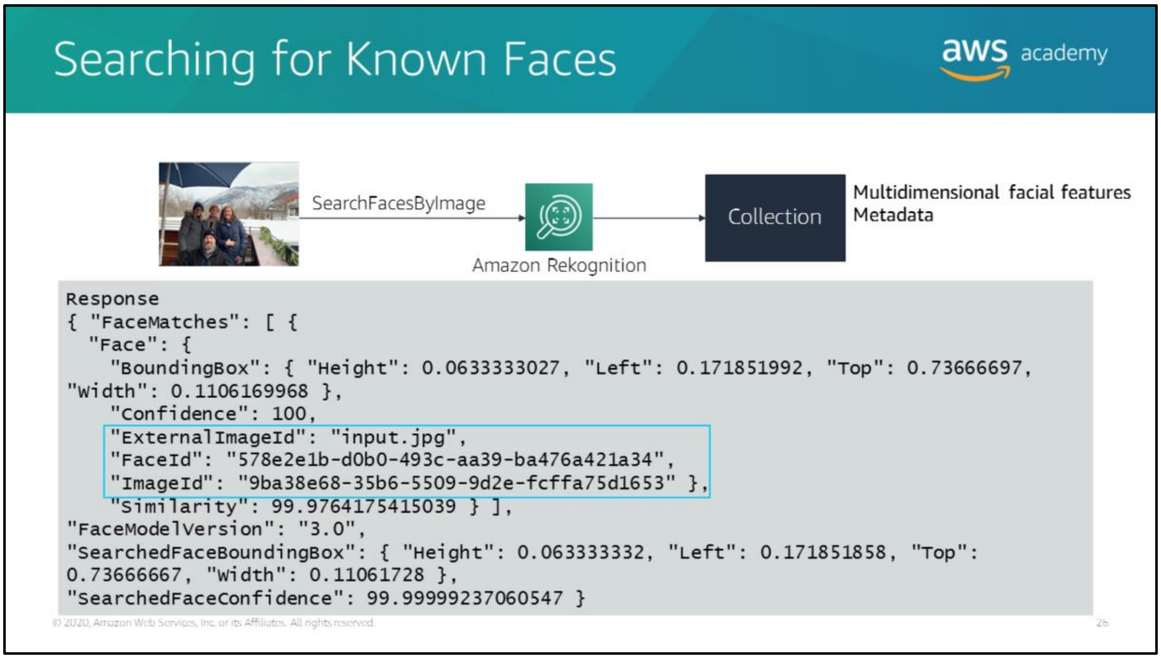
With Amazon Rekognition, you can compare two images to determine if they contain the same person. Comparisons require both a source and target image. The results will include all the faces that were found, and they include information about matching and non-matching faces. Again, confidence scores indicate how likely each prediction is.



Amazon Rekognition can also search for known faces. To use this feature, you must train the model by providing a collection of images to use. After you train the model, you can then detect those people in images that you provide.

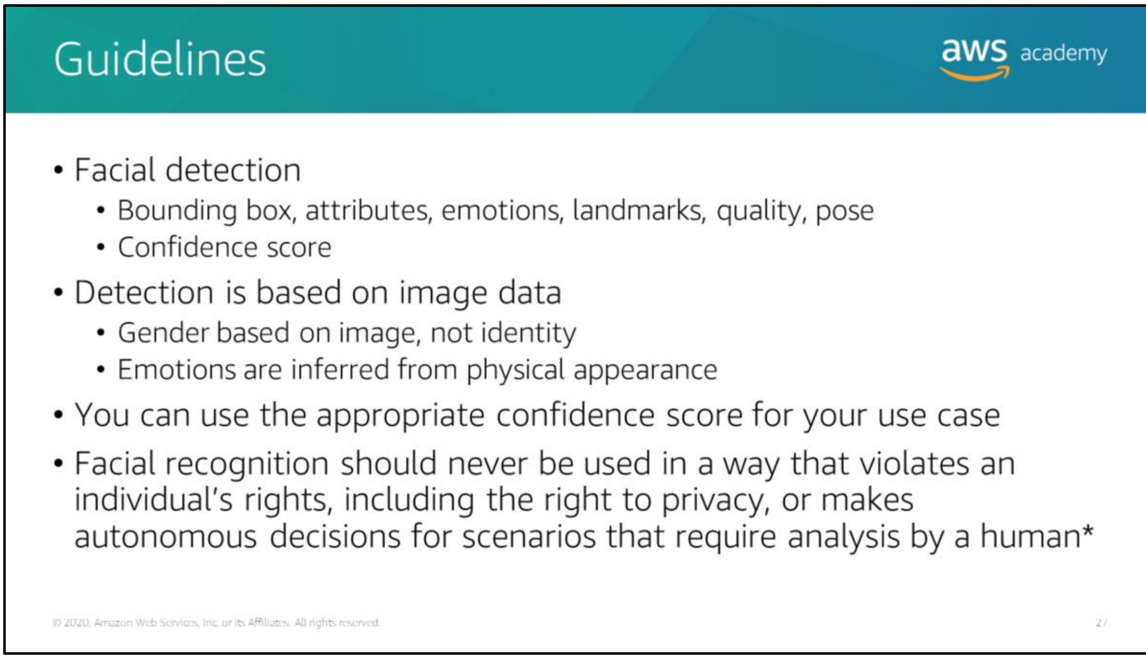
To find known faces, you must first create a collection and then add faces to the collection. Amazon Rekognition will perform facial recognition on the images that you provide and will return typical information like the bounding box coordinates or confidence score.

To associate faces with an image, you must specify an image ID in the ExternalImageId request parameter. This could be the file name of the image or another ID that you create.



After you create your collection, you can then use the SearchFacesByImage operation to search for faces from the collection.

The returned data contains an array of all faces that matched. The information includes bounding boxes, confidence scores, and the ExternalImageId value (which you can then use to link back to the source image).



When Amazon Rekognition detects a human face, it captures a bounding box that indicates where the face was found in the video. It also can detect attributes such as position of the eyes, nose, and mouth. It can detect emotion, quality of the detection, and any landmarks that might appear.

All these items will have an associated confidence score. A higher score indicates that the model has greater confidence about the detection.

Gender is inferred from the image. It is not inferred from identity. Similarly, emotion is also determined based on the image and it might not reflect the subject's actual emotional state.

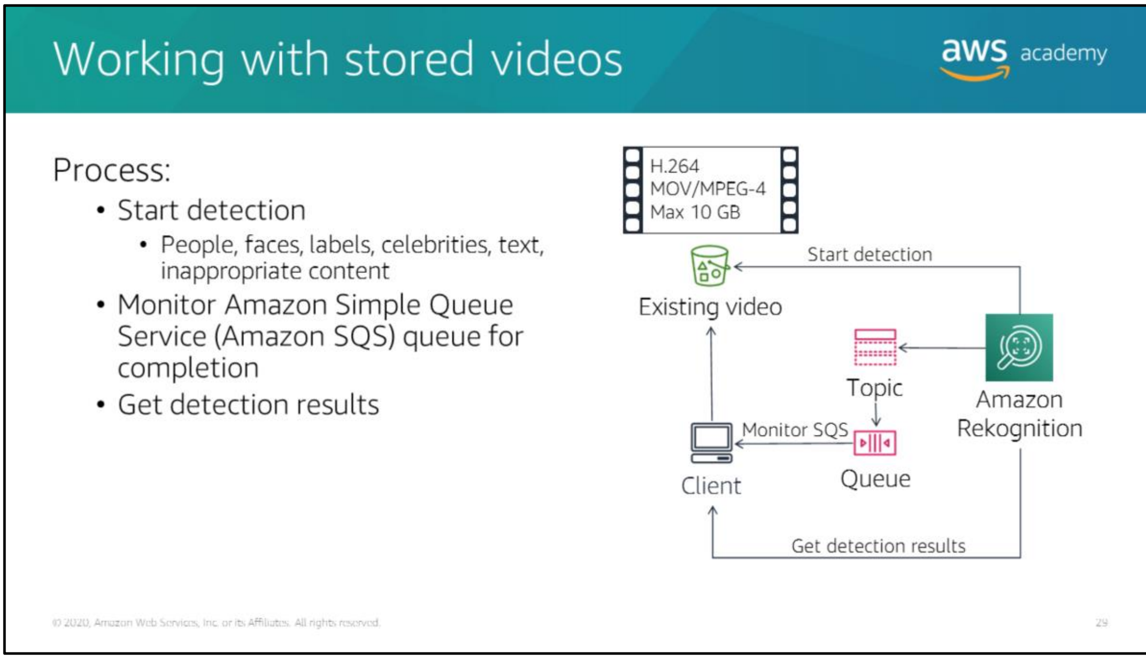
*** How should I apply facial recognition responsibly?**
Facial recognition should never be used in a way that violates an individual's rights, including the right to privacy. It should also never be used to make autonomous decisions for scenarios that require analysis by a human. For example, consider a bank that uses tools like Amazon Rekognition to verify their customers' identity in a financial application. The bank should always clearly disclose their use of the technology and ask the customer for their approval of the terms and conditions. With regard to public safety and law enforcement, we think that governments are free to work with law enforcement agencies to develop acceptable use policies for facial recognition technologies that both protect the rights of citizens and enable

law enforcement to protect the public's safety.

In all public safety and law enforcement scenarios, technology like Amazon Rekognition should only be used to narrow the field of potential matches. The responses from Amazon Rekognition enable officials to quickly get a set of potential faces for further human analysis. Given the seriousness of public-safety use cases, human judgment is needed to augment facial recognition, and facial recognition software should not be used autonomously.

For more information, refer to the AWS webpage: https://aws.amazon.com/rekognition/the-facts-on-facial-recognition-with-artificial-intelligence/

Your instructor will now either demonstrate Amazon Rekognition or provide you with access to a recorded demonstration.



You can perform video processing on both stored videos and video streams.

Stored videos should be stored in an S3 bucket.

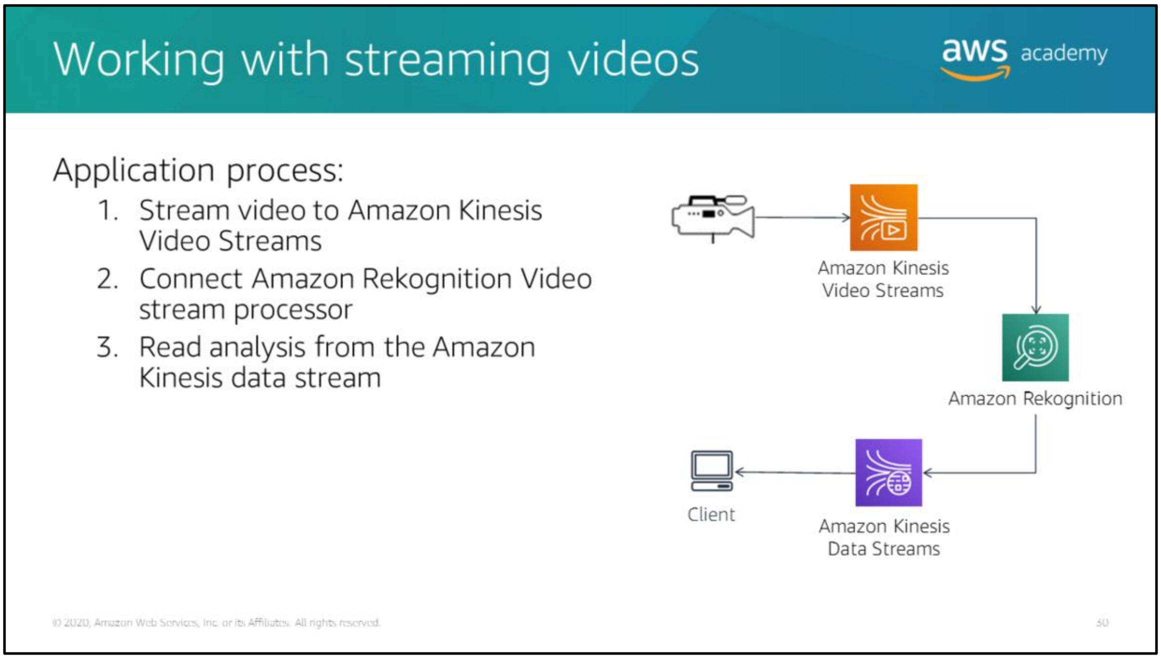
Each type of detection has its own Start operation. You can search for people, faces, labels, celebrities, text, and inappropriate content.

Amazon Rekognition publishes completion status to an Amazon Simple Notification Service (Amazon SNS) topic. Then, Amazon SNS can route these messages to subscribers. It is a best practice to route messages to an Amazon Simple Queue Service (Amazon SQS) message queue for durability. Your application should monitor the SQS queue for completion.

Each Start operation has a corresponding Get operation for retrieving the results.

- People: StartPersonTracking / GetPersonTracking
- Faces: StartFaceDetection / GetFaceDetection
- Labels: StartLabelDetection / GetLabelDetection
- Celebrities: StartCelebrityRecognition / GetCelebrityRecognition
- Explicit or suggestive adult content: StartContentModeration / GetContentModeration

Calling Get Detection Results returns an array of labels that contain information about any labels that were found in the video. The label information includes the same labels as image detection, along with a timestamp of where the label was detected in milliseconds from the start of the video.



You can use Amazon Rekognition Video to detect and recognize faces in streaming video. A typical use case is when you want to detect a known face in a video stream. Amazon Rekognition Video uses Amazon Kinesis Video Streams to receive and process a video stream. The analysis results are output from Amazon Rekognition Video to a Kinesis data stream and are then read by your client application. Amazon Rekognition Video provides a stream processor (CreateStreamProcessor) that you can use to start and manage the analysis of streaming video.

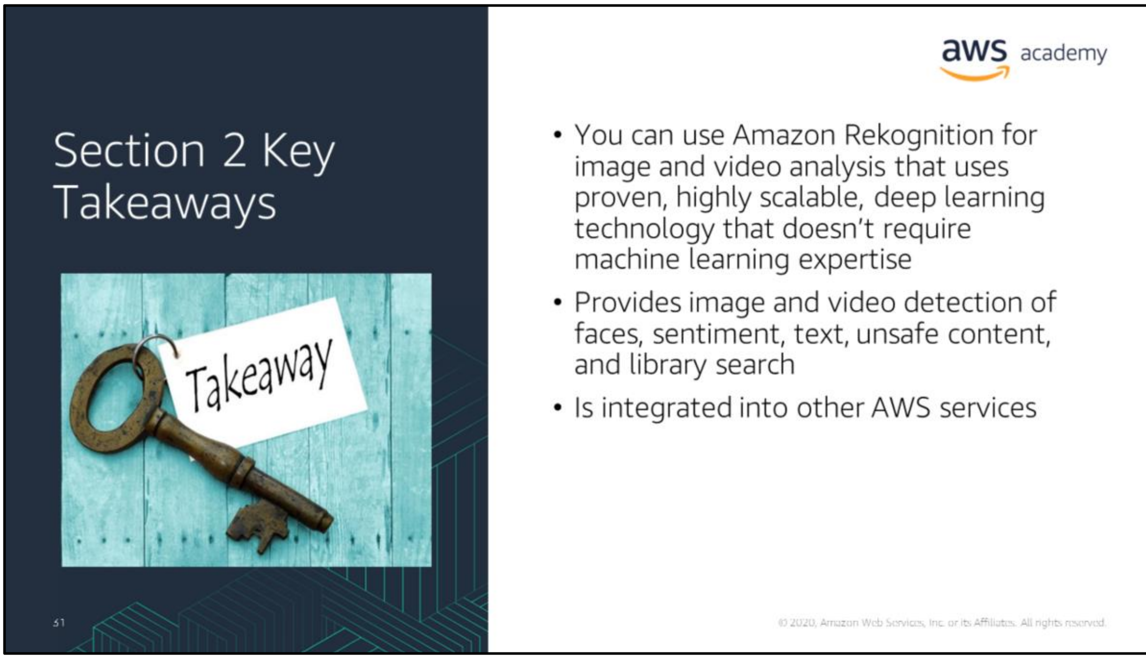
To use Amazon Rekognition Video with streaming video, your application must implement the following resources:

- A Kinesis video stream for sending streaming video to Amazon Rekognition Video
- An Amazon Rekognition Video stream processor to manage the analysis of the streaming video
- A Kinesis data stream consumer to read the analysis results that Amazon Rekognition Video sends to the Kinesis data stream

To find a face, you must create a collection. This process is the same as creating a collection when you work with images.

Amazon Rekognition Video places a JSON frame record for each analyzed frame into the Kinesis output stream. Amazon Rekognition Video doesn't analyze every frame that's passed to it through the Kinesis video stream.

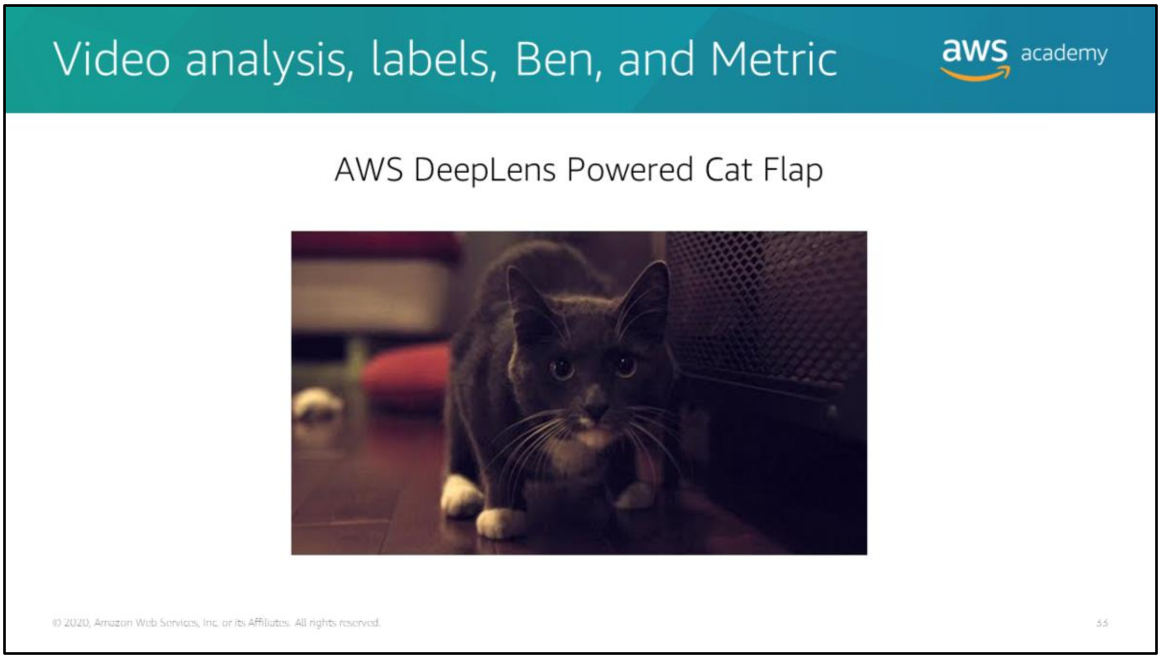
A frame record that's sent to a Kinesis data stream contains information about which Kinesis video stream fragment the frame is in, where the frame is in the fragment, and faces that are recognized in the frame. It also includes status information for the stream processor.



Some key takeaways from this section of the module include:

- Amazon Rekognition is a computer vision service that is based on deep learning. You can easily add image and video analysis to your applications,
- Amazon Rekognition provides image and video detection of faces, sentiment, text, unsafe content, and library search.
- Amazon Rekognition is integrated into other AWS services.

Introducing Section 3: Preparing custom datasets for computer vision.



You can watch these videos:

- [AWS DeepLens Powered Cat Flap]
- [Ben Hamm's Ignite presentation about the system that he created]

Models need training on domain

aws academy

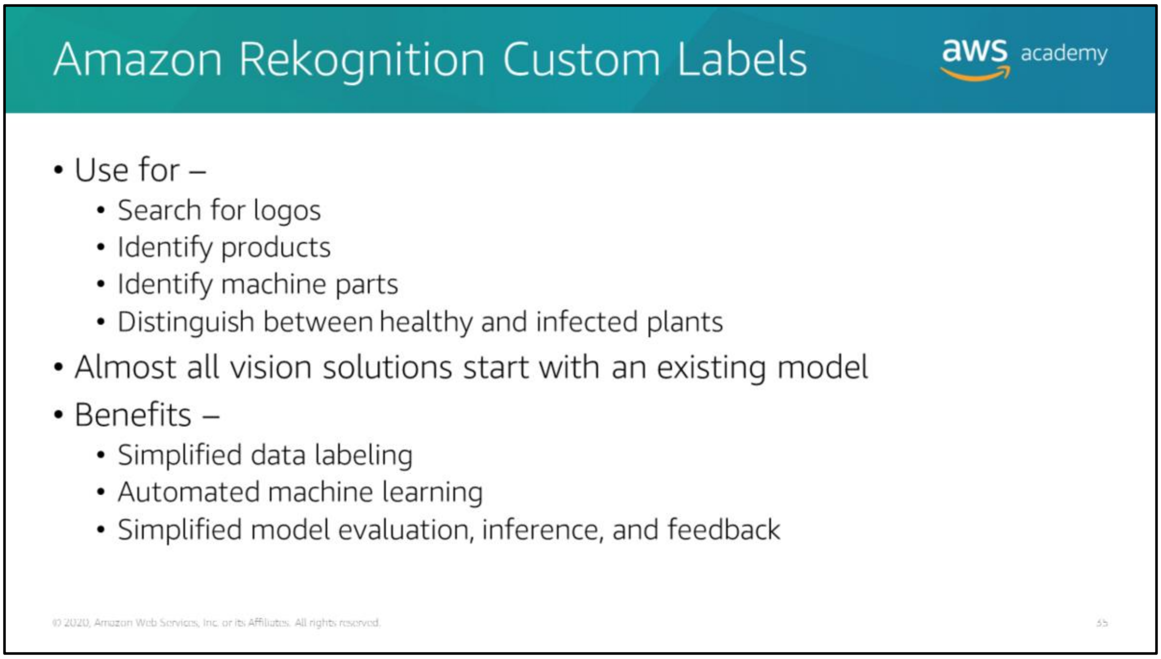
Wood	97%
Canvas	88.7%
Text	84.7%
Envelope	71.7%
Greeting Card	70.4%
Mail	70.4%
Hardwood	60.6%
Plywood	60.4%
Paper	57.7%
Advertisement	55.8%
Poster	55.8%
Art	55.6%

 54

One challenge of using a pre-built model is that it will only find images that it was trained to find. Though Amazon Rekognition was trained with tens of millions of images, it can't detect objects that it wasn't trained on.

For example, consider the eight of hearts playing card. If you run this card through Amazon Rekognition, the results show various attributes. However, none of the labels are *playing card* or *eight of hearts*. If you want Amazon Rekognition to detect images in your problem domain, you must train the model with your images.

In this section, you will learn how to train Amazon Rekognition with images from your problem domain. Though you will learn about Amazon Rekognition specifically, the process will be similar if you use other pre-trained models.



As with other machine learning processes, you must train Amazon Rekognition to recognize scenes and objects that are in a domain. Thus, you need a training dataset and a test dataset that contains labeled images. Amazon Rekognition Custom Labels can be helpful for these tasks.

You can use Amazon Rekognition Custom Labels to find objects and scenes that are unique to your business needs. For example, you can use it to classify images (image-level predictions) or detect images (object-level or bounding-box-level predictions). For example, if you want to identify specific machine parts, such as a turbocharger or a torque converter, you could collect pictures of each part and use them to train a model.

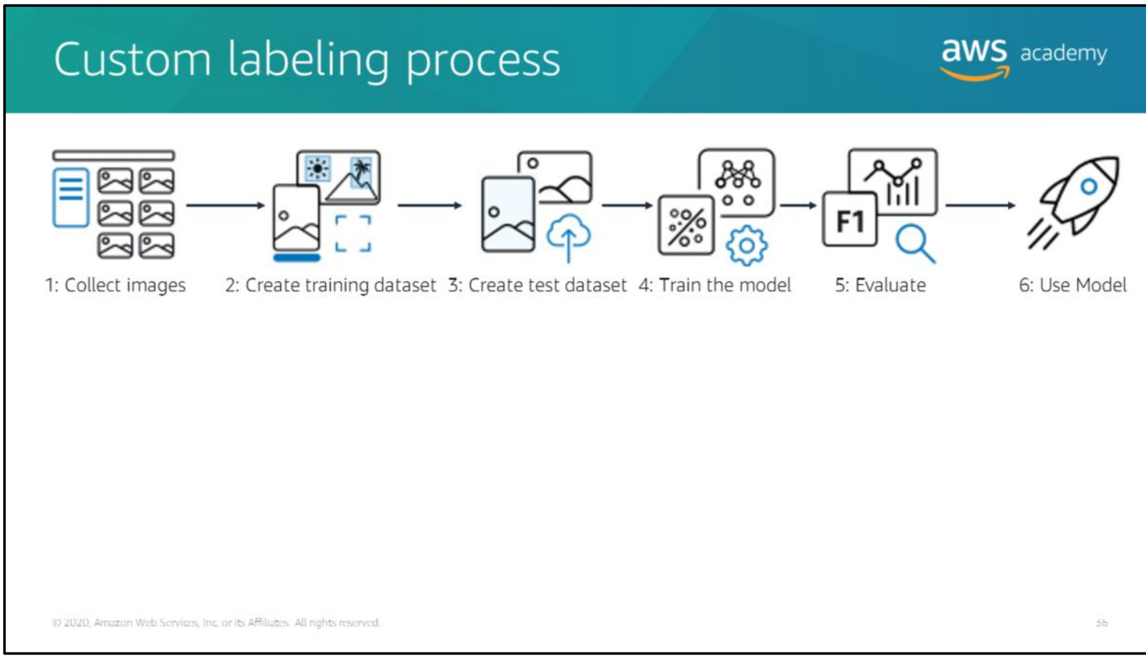
Many machine learning problems today can be solved by training existing models. Training a computer vision algorithm to recognize images requires a large input dataset, which is impractical for most organizations.

You can use an existing model or a managed service like Amazon Rekognition Custom Labels to:

- **Simplify data labeling** – Amazon Rekognition Custom Labels provides a UI for labeling images, including defining bounding boxes.

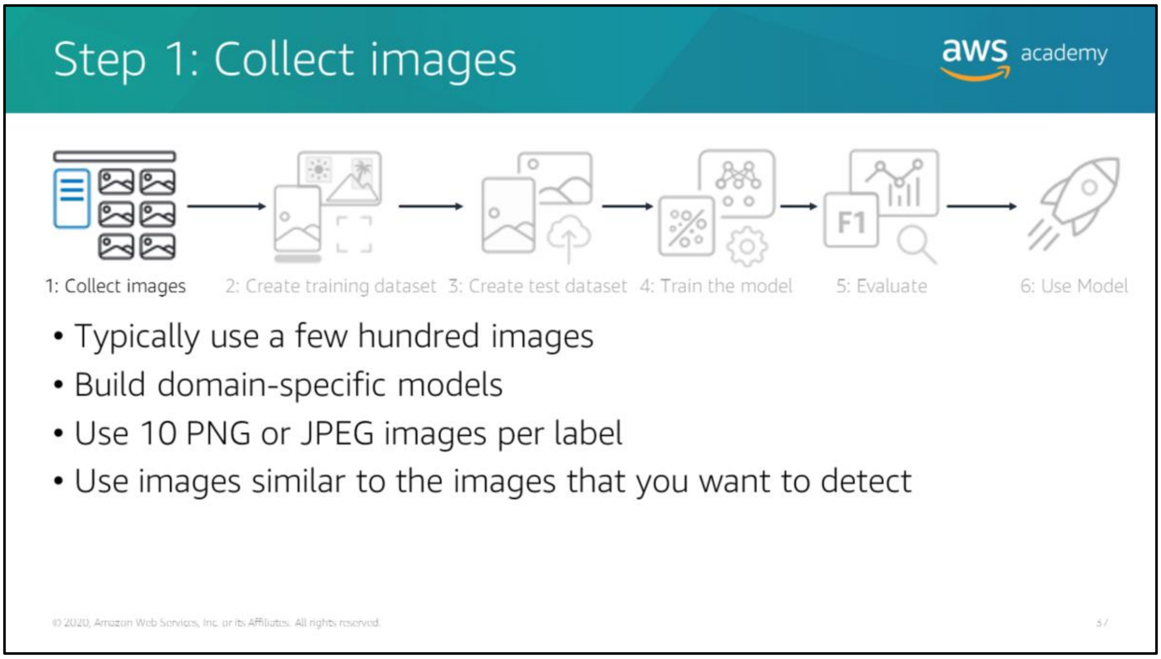
- **Provide automated machine learning** – Amazon Rekognition Custom Labels includes automated machine learning capabilities that handle the ML process for you. When you provide training images, Amazon Rekognition Custom Labels can automatically load and inspect the data, select the correct machine learning algorithms, train a model, and provide model performance metrics.
- **Provide simplified model evaluation, inference, and feedback** – You evaluate your custom model's performance on your test set. For every image in the test set, you can see the side-by-side comparison of the model's prediction versus the label that it assigned. You can also review detailed performance metrics. You can start using your model immediately for image analysis, or you can iterate and retrain new versions with more images to improve performance.

After you start using your model, you can track your predictions, correct any mistakes, and use the feedback data to retrain new model versions and improve performance.



The diagram shows a typical process for training a computer vision model, which includes the Amazon Rekognition Custom Labels feature.

Step 1 – Collect images that contain the objects or scenes you want to find.
Step 2 – Upload and label images from your computer or Amazon S3, or import an Amazon SageMaker Ground Truth .manifest file for already labeled images.
Step 3 – Create a dataset to evaluate your model's performance, select an existing dataset, or split your training dataset for testing.
Step 4 - Train your custom model by using your training datasets. The best ML techniques will automatically be selected.
Step 5 - Evaluate your model performance on your test dataset. Improve your model by adding images to the training dataset.
Step 6 - Use your customer model to analyze images with an API operation.



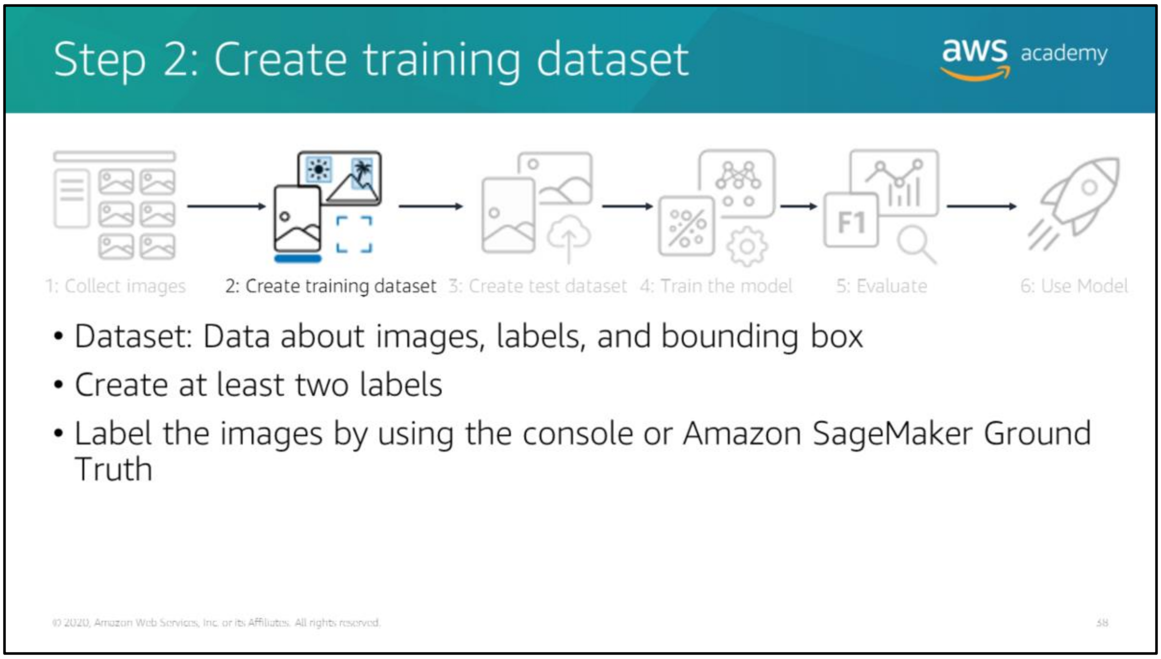
The process of developing a custom model to analyze images requires time, expertise, and resources. It often takes months to complete. Additionally, it can require thousands or tens of thousands of hand-labeled images to provide the model with enough data to make decisions accurately. It can take months to generate and gather this data, and it can require large teams of labelers to prepare it for use in machine learning.

Amazon Rekognition Custom Labels builds on the existing capabilities of Amazon Rekognition, which are already trained on tens of millions of images across many categories. Instead of thousands of images, you can upload a small set of training images that are specific to your use case. You typically use a few hundred images or less. You can upload training images by using the console. If your images are already labeled, Amazon Rekognition Custom Labels can begin training a model in a short time. If not, you can label the images directly in the labeling interface, or you can use Amazon SageMaker Ground Truth to label them for you.

Amazon Rekognition Custom Labels works best when you use different models for different domains. For example, if you must detect machine parts and plant health, you would use two different models.

Images that you select for training should be similar to the images that will be used for inference. You should use images that use various lighting conditions, backgrounds, and resolutions ideally that mirror images that you would want to perform detection on. If you can use the same source, as you would use in production, that works best.

The documentation includes additional guidelines on image type (JPEG or PNG), image size, resolution, and other properties.



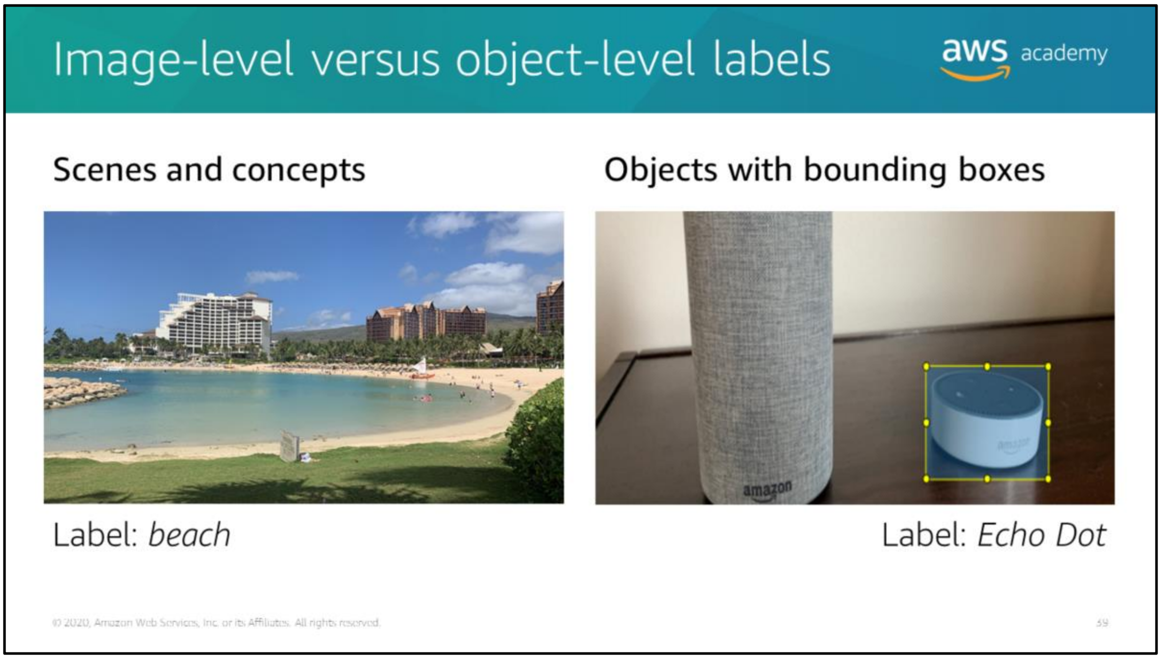
Datasets contain information about images, labels, and bounding boxes that is needed to train and test an Amazon Rekognition Custom Labels model.

You can use images from Amazon S3, or upload them from your computer to Amazon S3 as part of the process.

To train a model, your dataset should have at least two labels, with at least 10 images per label. Each image in your dataset must be labeled. You can use the Amazon Rekognition Custom Labels console or use Amazon SageMaker Ground Truth.

Amazon SageMaker Ground Truth enables you to use workers from your own workforce, Amazon Mechanical Turk, or a vendor company to label the images. You can import the results of these labeling efforts into an Amazon Rekognition Custom Labels dataset.

If you have images that are already labeled, you must convert the labels into an Amazon SageMaker Ground Truth manifest file. To learn more details about the format, refer to https://docs.aws.amazon.com/rekognition/latest/customlabels-dg/cd-manifest-files.html



To train an Amazon Rekognition Custom Labels model, your images must be labeled. A label indicates that an image contains an object, scene, or concept.

A dataset needs at least two defined labels. Each image must have at least one assigned label that identifies the object, scene, or concept in the image.

Scenes and concepts

You can apply labels to an image as a whole. These labels are known as image-level labels. They are useful for identifying scenes or concepts that you want to detect. For example, the following image shows a beach scene from Ko Olina from the island of Oahu in the US state of Hawaii. To train a model to detect beaches, you would add a *beach* label that applies to the entire image.

Objects with bounding boxes

Labels can be applied to areas of an image that contain an object that you want to detect. For example, if you want your model to detect Amazon Echo devices, it must identify the types of Echo Devices that are in an image. The model needs information about where the devices are in the image, and a corresponding label that identifies the type of the device. This information is known as localization information. The location of the device is expressed as a bounding box. The example *Objects with bounding boxes* image shows a bounding box that surrounds an Amazon Echo Dot. The image also contains an Amazon Echo without a bounding box.

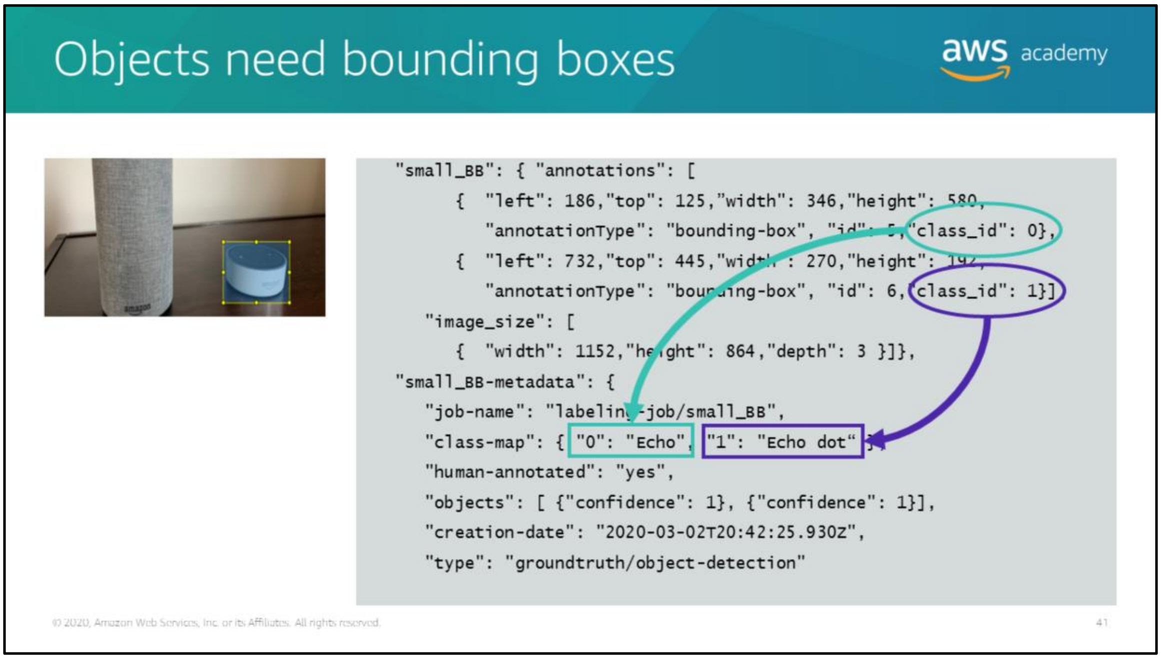
Images need labels

aws academy

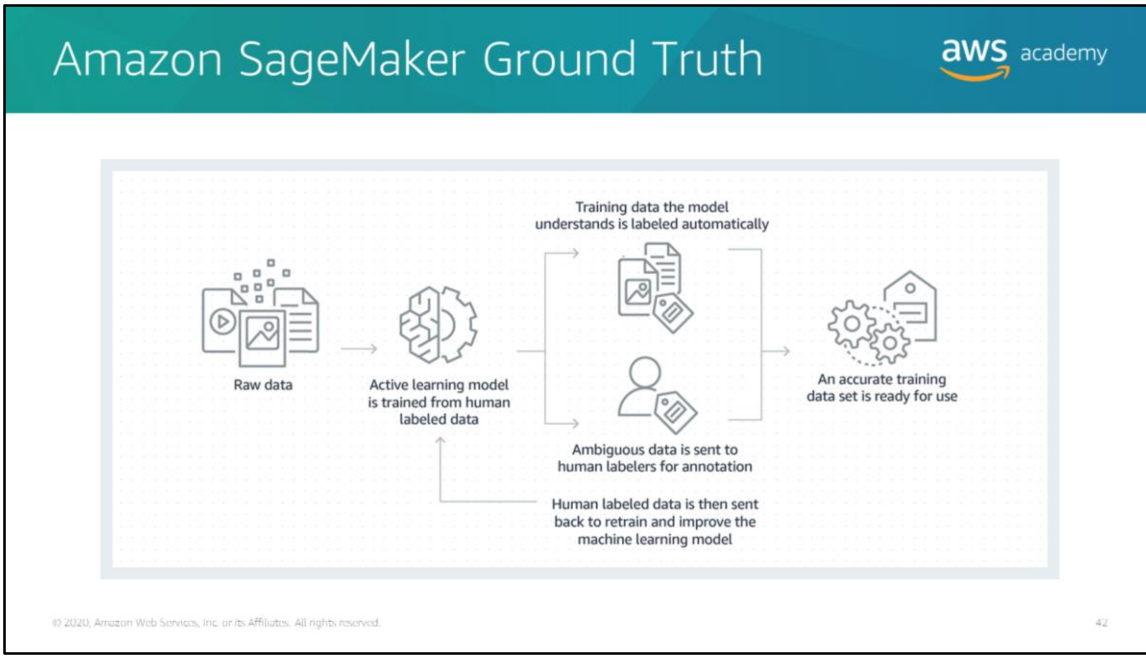
```
{
    "source-ref": "s3://b/…/img_2783.jpg",
    "small_beach": 1,
    "small_beach-metadata": {
       "confidence": 1,
       "job-name": "labeling-job/small_beach",
       "class-name": “Beach",
       "human-annotated": "yes",
       "creation-date": "2020-03-02T20:42:03.525Z",
       "type": "groundtruth/image-classification"
    }
}
```

 40

The manifest file for an image level label will typically contain the label, or *class-name*, along with some metadata about how the image was labeled.



For object detection, the manifest will contain information about each labeled image. The bounding box identifies where the object is in the image, along with the label that bounding box belongs to.



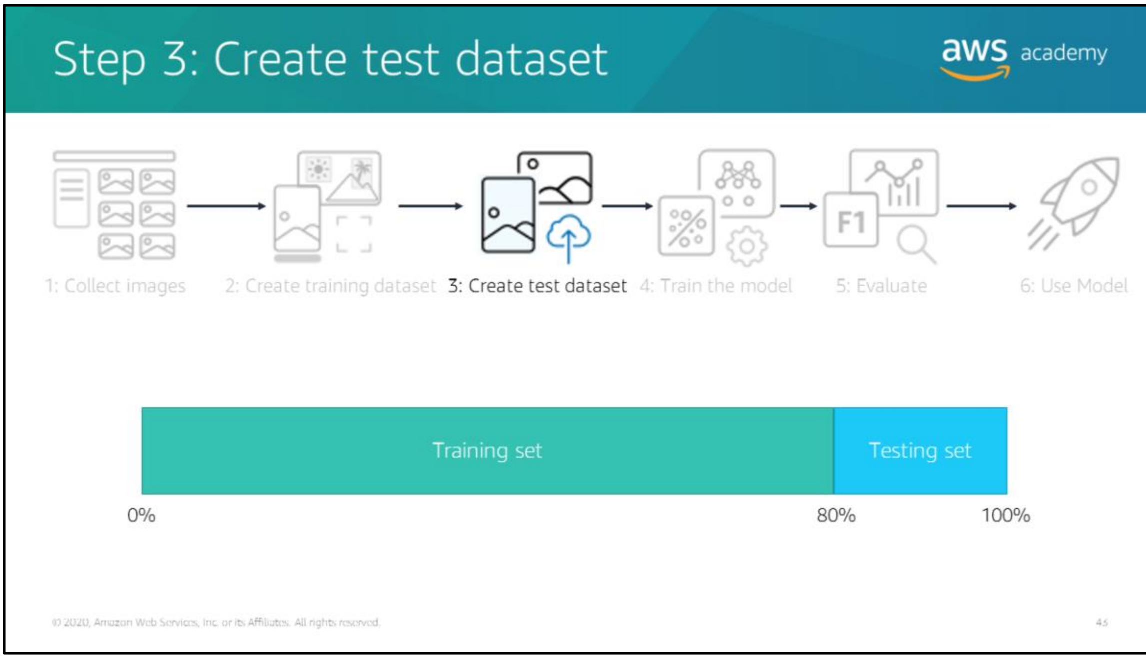
Amazon SageMaker Ground Truth enables you to build high-quality training datasets for your machine learning models. As mentioned earlier, you can use workers from Amazon Mechanical Turk, a vendor company, or an internal workforce with machine learning to create a labeled dataset. You can use the labeled dataset output from Ground Truth to train your own models. You can also use the labeled dataset for Amazon Rekognition Custom Labels.

Ground Truth can use active learning to automate the labeling of your input data. *Active learning* is a machine learning technique that identifies data that should be labeled by your workers. In Ground Truth, this functionality is called automated data labeling. Automated data labeling can reduce the cost and time that it takes to label your dataset compared to using only humans. When you use automated labeling, you incur Amazon SageMaker training and inference costs.

We recommend using automated data labeling on large datasets. The neural networks that are used with active learning need a large amount of data for every new dataset. Typically, the potential for highly accurate predictions increases when more data is provided. Data will only be automatically labeled if the neural network that is used in the automated labeling model can achieve an acceptably high level of accuracy. Data will be automatically labeled only if the neural network that's used in the automated labeling model can achieve an acceptably high level of accuracy. With a larger dataset, the neural network has more potential to achieve high-enough level accuracy for automatic labeling. Automated data labeling is most appropriate when you have thousands of data objects. The minimum number of objects allowed for automated data labeling is 1,250. However, we strongly suggest providing a minimum of 5,000 objects.

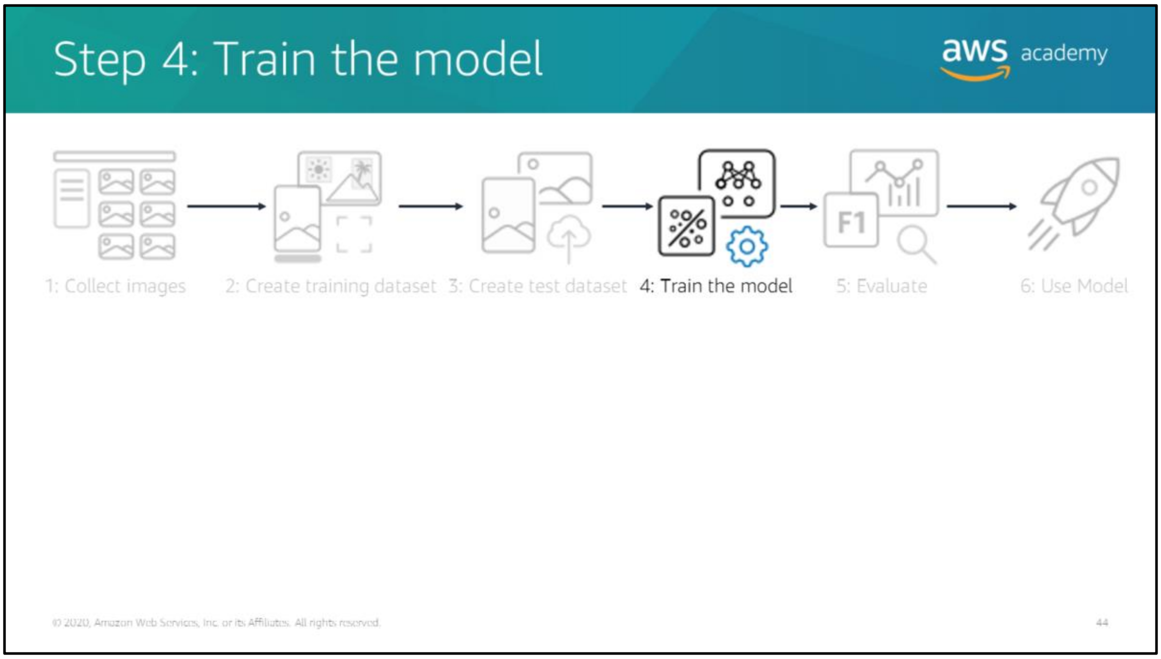
You enable automated data labeling when you create a labeling job. This process outlines how it works:

1. When Ground Truth starts an automated data labeling job, it selects a random sample of input data (objects) and sends it to human workers.
2. When the labeled data is returned, Ground Truth uses this data (the validation data) to validate the models that were trained for automated data labeling.
3. Ground Truth runs a batch transform job by using the validated model for inference on the validation data. Batch inference produces a confidence score and quality metric for each object in the validation data.
4. The automated labeling component uses these quality metrics and confidence scores to create a confidence score threshold that is designed to support label quality.
5. Ground Truth runs a batch transform job on the unlabeled data in the dataset by using the same validated model for inference. This job will produce a confidence score for each object.
6. The Ground Truth automated labeling component determines if the confidence score for each object (which was produced in step 5) meets the required threshold (which was determined in step 4). If the confidence score meets the threshold, the expected quality of automatic labeling exceeds the requested level of accuracy. That object is then considered to be automatically labeled.
7. Step 6 produces a dataset of unlabeled data with confidence scores. Ground Truth selects data points with low confidence scores from this dataset, and sends them to human workers for additional labeling.
8. Ground Truth uses the existing human-labeled data and the additional human-labeled data to train a new model.
9. The process is repeated until the dataset is fully labeled or until another stopping condition is met. For example, automatic labeling can stop when you meet your budget for human annotation.



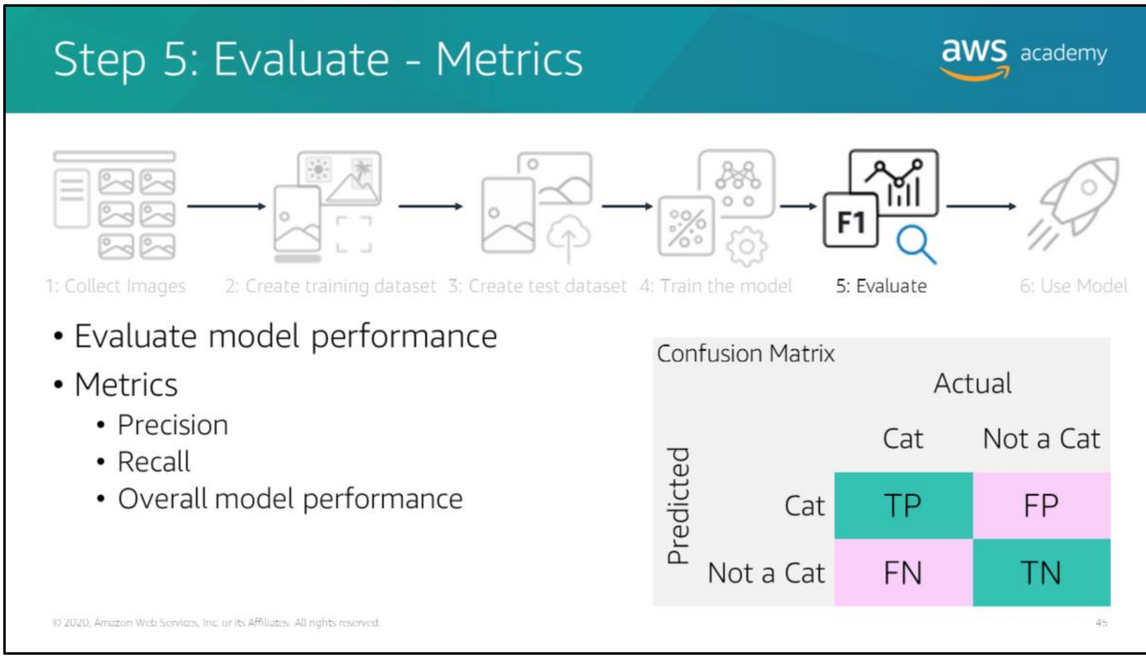
The final step before you train your model is to identify a test dataset. You will use this test dataset to validate and evaluate the model's performance by performing an inference on the images that are in the test dataset. You will then compare the results with the labeling information that is in the training dataset.

You can create your own test dataset. Alternatively, you can use Amazon Rekognition Custom Labels to split your training dataset into two datasets by using an 80/20 split. This split means that 80 percent of the data is used for training and 20 percent of the data is used for testing.



After you define the training and test datasets, Amazon Rekognition Custom Labels can automatically train the model for you. Amazon Rekognition Custom Labels automatically loads and inspects the data, selects the correct machine learning algorithms, trains a model, and provides model performance metrics.

You are charged for the amount of time that a model takes to train. A dataset that contains more images and labels will take longer to train.



When training is complete, you evaluate the performance of the model.

During testing, Amazon Rekognition Custom Labels predicts if a test image contains a custom label. The confidence score is a value that quantifies the certainty of the model's prediction.

Since this is a classification problem, the results can be mapped to a confusion matrix:

- *True positive (TP)* – The Amazon Rekognition Custom Labels model correctly predicts the presence of the custom label in the test image. That is, the predicted label is also a *ground truth* label for that image. For example, Amazon Rekognition Custom Labels correctly returns a *cat* label when a cat is present in an image.
- *False positive* (FP) – The Amazon Rekognition Custom Labels model incorrectly predicts the presence of a custom label in a test image. That is, the predicted label isn't a ground truth label for the image. For example, Amazon Rekognition Custom Labels returns a *cat* label, but there is no cat label in the ground truth for that image.
- *False negative* (FN) – The Amazon Rekognition Custom Labels model doesn't predict that a custom label is present in the image, but the ground truth for that image includes this label. For example, Amazon Rekognition Custom Labels doesn't return a *cat* custom label for an image that contains a cat.
- *True negative* (TN) – The Amazon Rekognition Custom Labels model correctly predicts that a custom label isn't present in the test image. For example, Amazon Rekognition

Custom Labels doesn’t return a *cat* label for an image that doesn’t contain a cat.

The console provides access to true positive, false positive, and false negative values for each image in your test dataset.

These prediction results are used to calculate the various metrics for each label and an aggregate of metrics for your entire test set. The same definitions apply to predictions that the model makes at the bounding-box level. With bounding boxes, all metrics are calculated over each bounding box (whether prediction or ground truth) in each test image.

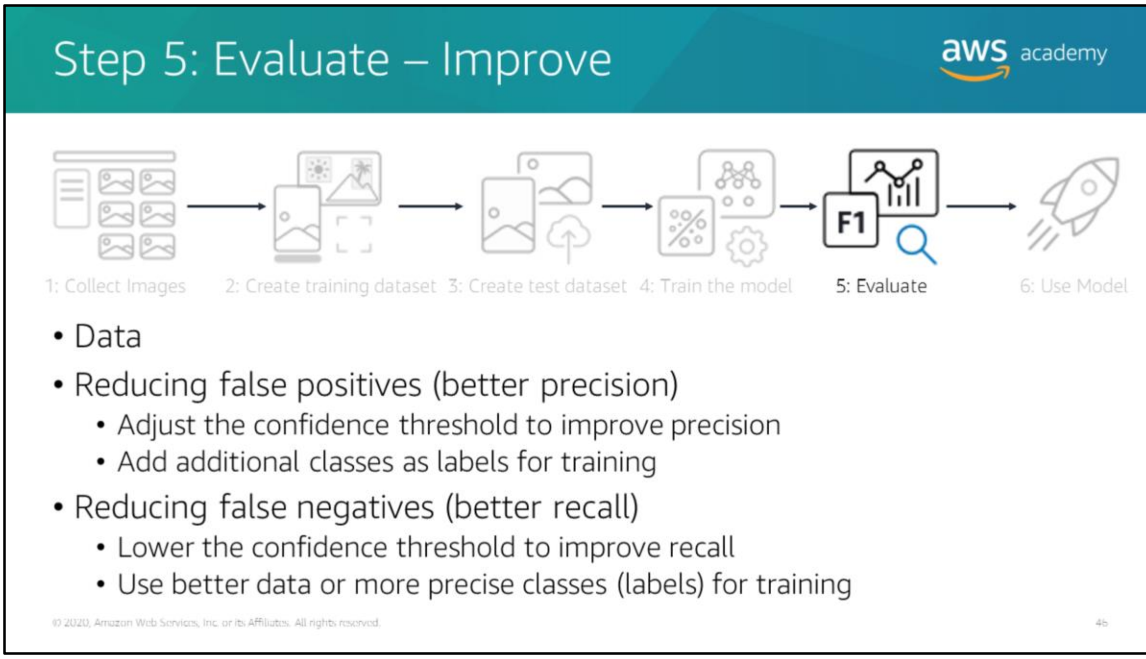
To help you, Amazon Rekognition Custom Labels provides summary metrics and evaluation metrics for each label.

Precision metrics
Amazon Rekognition Custom Labels provides precision metrics for each label and an average precision metric for the entire test dataset.
Precision is the proportion of positive results that were correctly classified

Recall metrics
Amazon Rekognition Custom Labels provides average recall metrics for each label and an average recall metric for the entire test dataset.
Recall is the fraction of your test set labels that were correctly classified. In our example that would be how many cats were correctly classified.

Overall model performance
Amazon Rekognition Custom Labels provides an average model performance score for each label and an average model performance score for the entire test dataset. The F1 Score combines precision and recall together to give you just one number to quantify the overall performance of a particular ML algorithm. You should look at using F1 score when you have a class imbalance but want to preserve the equality between precision and sensitivity. A higher value means better model performance for both recall and precision.

If you're satisfied with the accuracy of your model, you can start to use it.



Data

In general, you can improve the quality of your model with larger quantities of better-quality data. Use training images that clearly show the object or scene and don't include many things that you are not interested in. For bounding boxes around objects, use training images that show the object fully visible and not hidden by other objects.

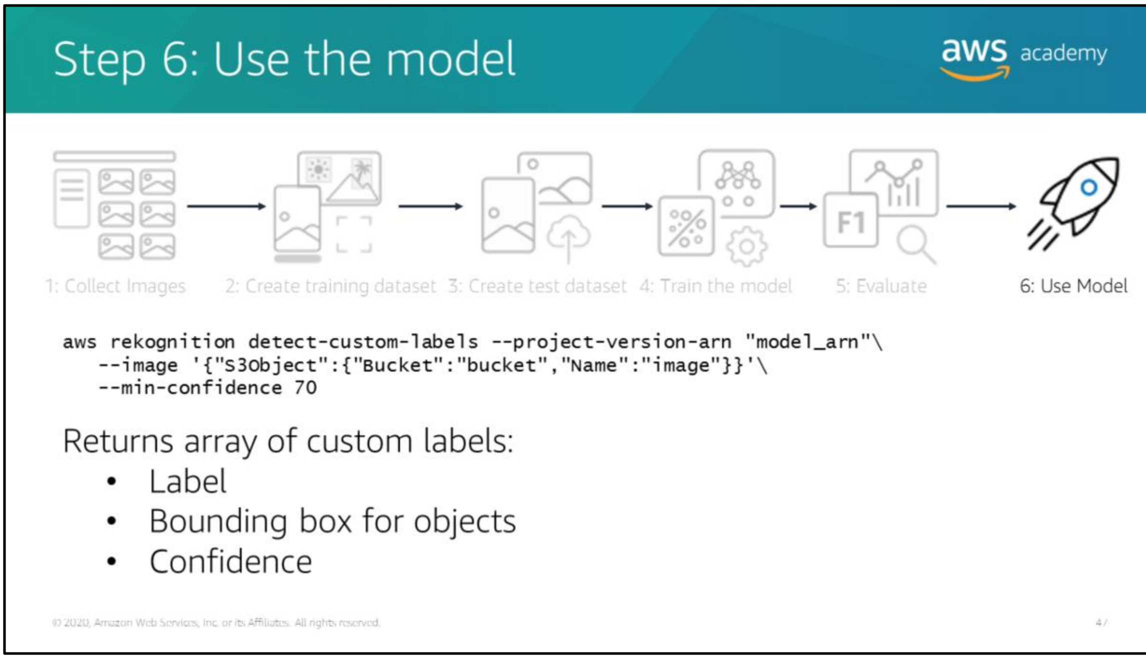
Make sure that your training and test datasets match the type of images that you will eventually run inference on. For objects, where you have just a few training examples (like logos), you should provide bounding boxes around the logo in your test images. These images represent the scenarios that you want to localize the object in.

Reducing false positives (better precision)

- First, check if increasing the confidence threshold enables you to keep the correct predictions while eliminating false positives. Increasing the confidence threshold eventually results in diminishing gains because of the tradeoff between precision and recall for a given model.
- You might see one or more of your custom labels of interest (A) consistently get confused with the same class of objects (but not a label that you're interested in) (B). To help, add B as an object class as a label to your training dataset (along with the images that you got the false positive on). Effectively, you're helping the model learn to predict B and not A

through the new training images.
- You might find that the model is confused between two of your custom labels (A and B)—the test image with label A is predicted as having label B and vice versa. In this case, first check for mislabeled images in your training and test sets. Also, adding more training images that reflect this confusion will help a retrained model learn to better discriminate between A and B.

Reducing false negatives (better recall)
- Lower the confidence threshold to improve recall.
- Use better examples to model the variety of both the object and the images in which they appear.
- Split your label into two classes that are easier to learn. For example, instead of *good cookies* and *bad cookies*, you might want *good cookies*, *burnt cookies*, and *broken cookies* to help the model better learn each unique concept.



If you are satisfied with the performance of your model, you can make it available for use by starting it from the console or by using code.

After the model is running, you can perform an inference by using the AWS CLI or the SDK.

When you call the API, you specify:

- The Amazon Resource Name (ARN) of the Amazon Rekognition Custom Labels model that you want to use.
- The image that you want the model to make a prediction with. You can provide an input image as an image byte array (base64-encoded image bytes) or as an Amazon S3 object.

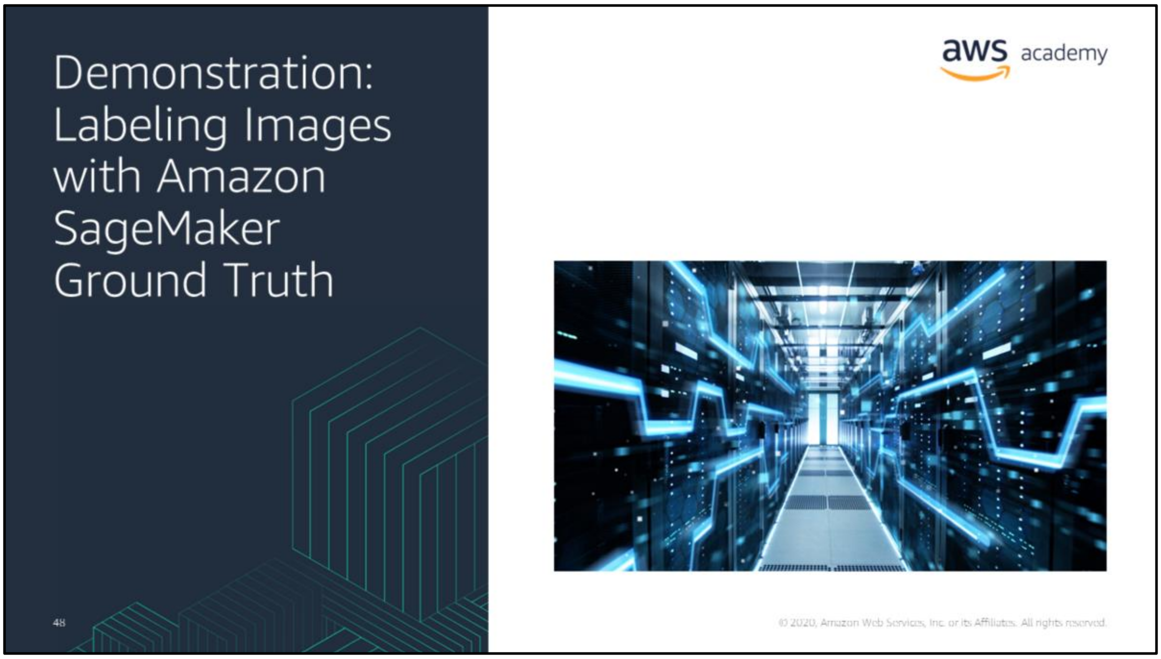
Custom labels are returned in an array of Custom Label objects. Each custom label represents a single object, scene, or concept that is found in the image. A custom label includes:

- A label for the object, scene, or concept that was found in the image.
- A bounding box for objects that were found in the image. The bounding box coordinates show where the object is located on the source image. The coordinate values are a ratio of the overall image size.
- The confidence that Amazon Rekognition Custom Labels has in the accuracy of the label

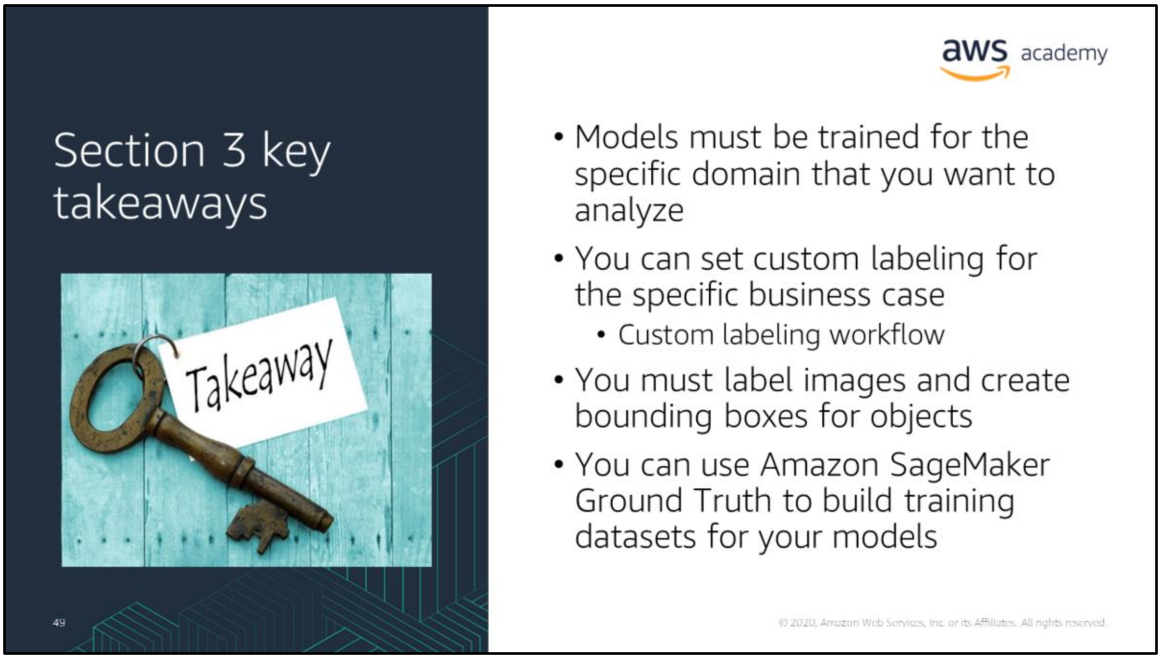
and bounding box.

During training, a model calculates a threshold value that determines if a prediction for a label is true. By default, the DetectCustomLabels operation doesn't return labels with confidence value that is less than the model's calculated threshold value. To filter the labels that are returned, specify a value for MinConfidence that is greater than the model's calculated threshold. You can get the model's calculated threshold from the model's training results shown in the Amazon Rekognition Custom Labels console. To get all the labels regardless of confidence, specify a MinConfidence value of *0*.

If you find that the confidence values returned by the DetectCustomLabels operation are too low, consider retraining the model. You can restrict the number of custom labels that are returned from the DetectCustomLabels operation by specifying the MaxResults input parameter. The returned results are sorted from the highest confidence to the lowest confidence.



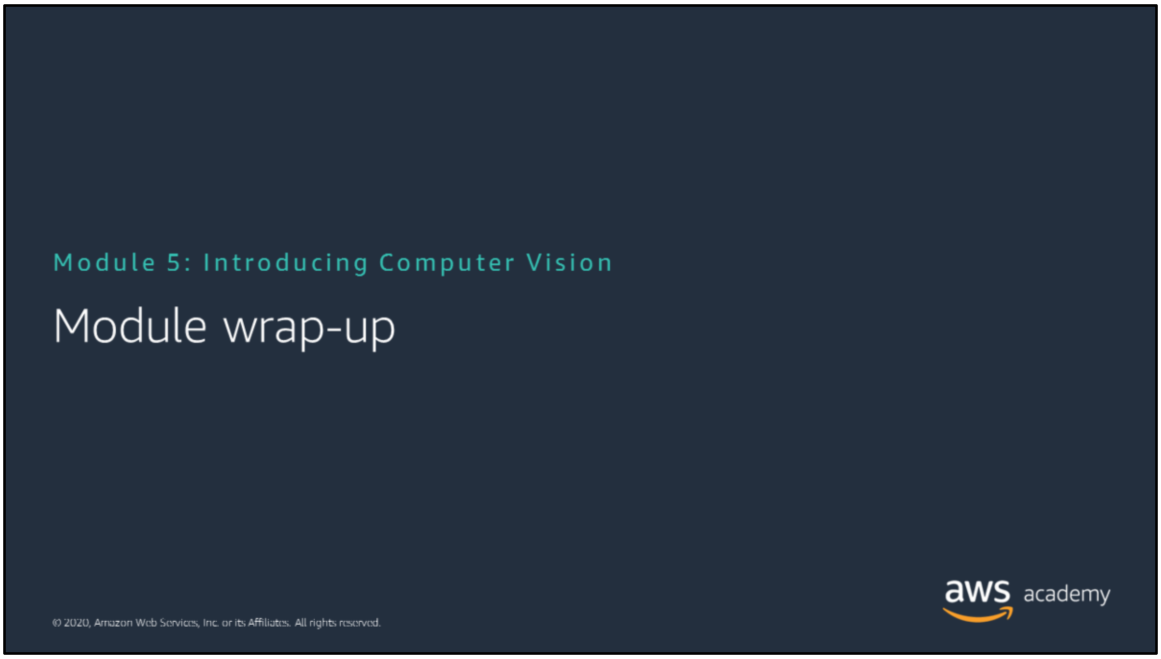
Your instructor will now either demonstrate how to label images with Amazon SageMaker Ground Truth or provide you with access to a recorded demonstration.



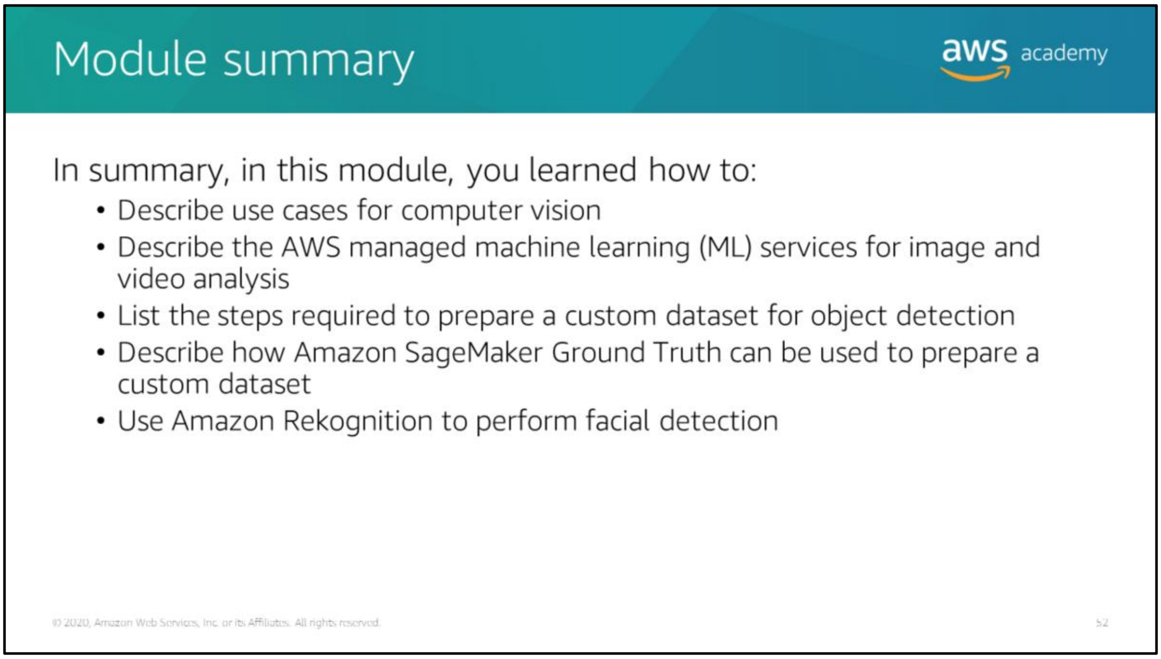
Some key takeaways from this section of the module include:

- Models must be trained for the specific domain that you want to analyze
- You can set custom labeling for the specific business case
    - Custom labeling workflow
- You must label images and create bounding boxes for objects
- You can use Amazon SageMaker Ground Truth to build training datasets for your models

You will now complete Module 5 – Guided Lab: Facial Recognition.

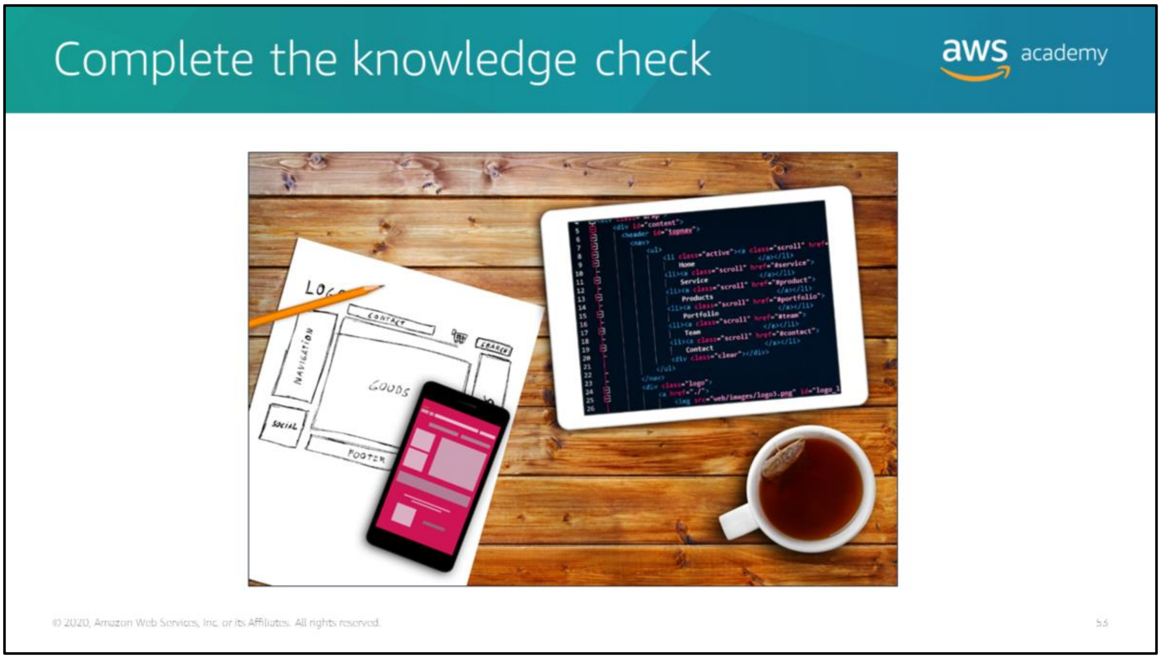


It’s now time to review the module and wrap up with a knowledge check.



In summary, in this module, you learned how to:

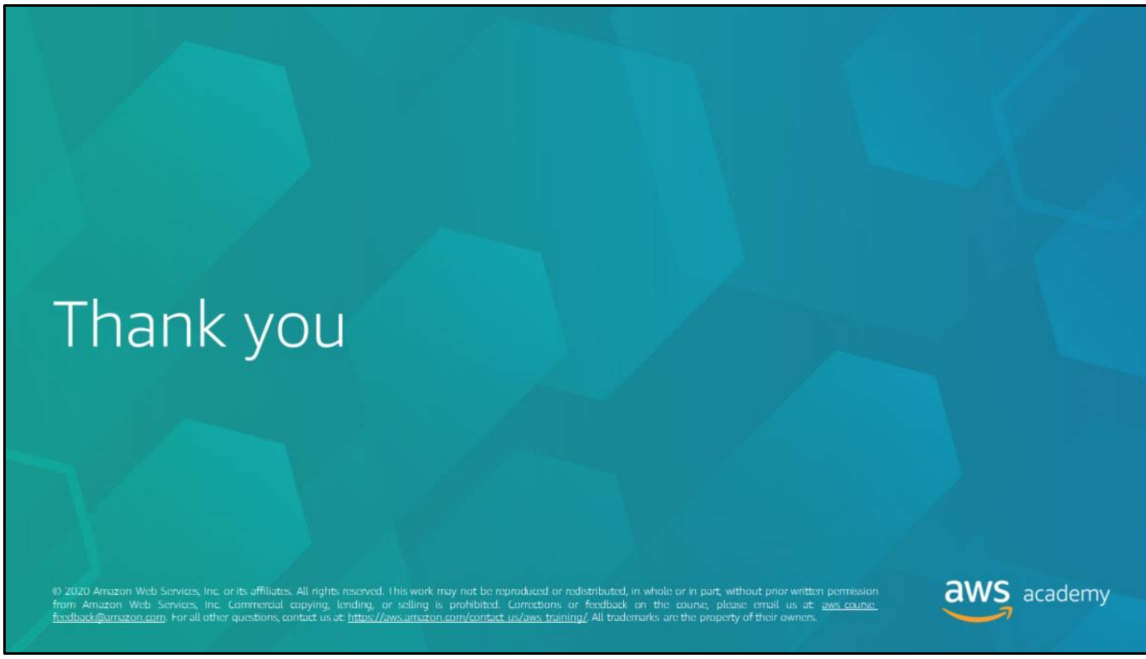
- Use cases for computer vision
- Describe the AWS managed machine learning (ML) services for image and video analysis
- List the steps required to prepare a custom dataset for object detection
- Describe how Amazon SageMaker Ground Truth can be used to prepare a custom dataset
- Use Amazon Rekognition to perform facial detection



It is now time to complete the knowledge check for this module.

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- What is Amazon Rekognition
- Welcoming Amazon Rekognition Video: Deep-Learning Based Video Recognition
- Classify a Large Number of Images with Amazon Rekognition and AWS Batch



Thanks for participating.