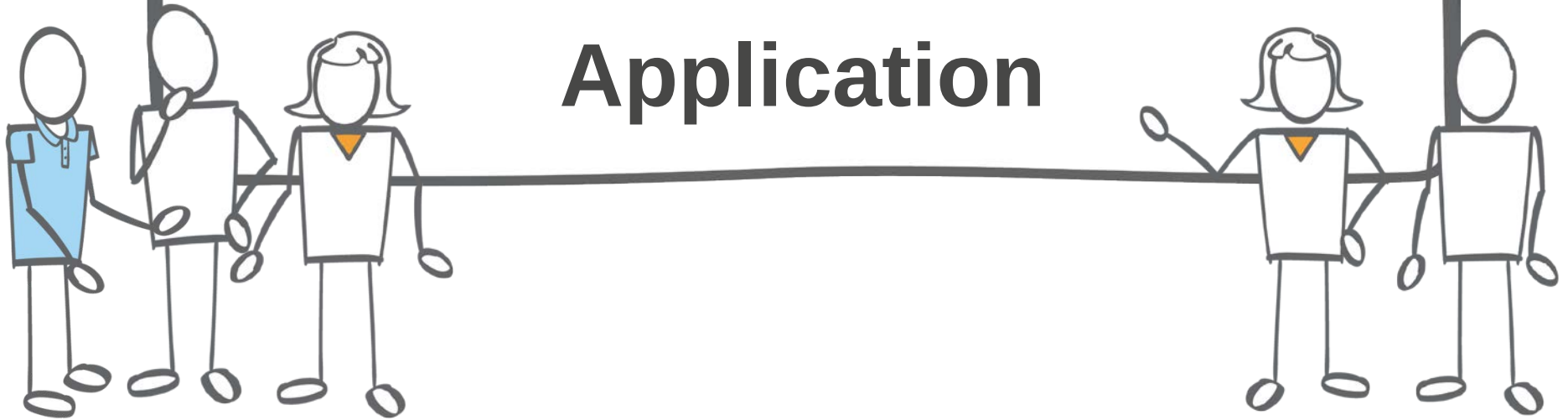
Build a Serverless, Location-Aware, Search & Recommendations-Enabled Application



What You will be Building Today

- A food delivery service
- An interactive mapping user interface
- A serverless web application and backend infrastructure with API Gateway and Lambda
- Predictive recommendations with AWS Machine Learning

Modules

Module1: Designing and Building the API Backend

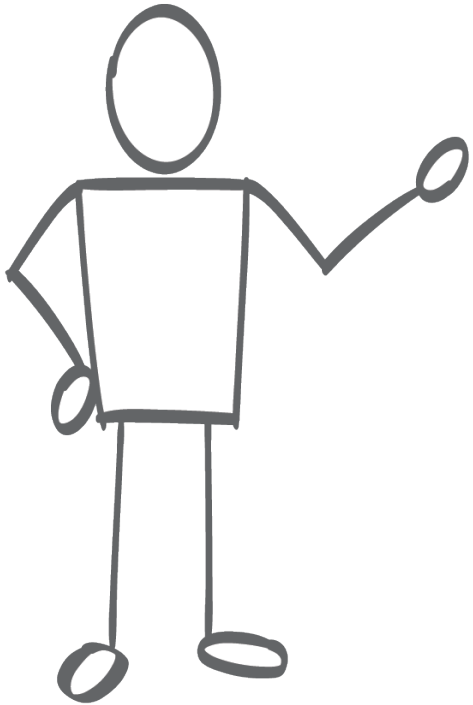
Module 2: Building a Serverless Web Application

Module 3: An Introduction to Data Science Using Amazon Machine Learning

Demo

What you will build today…

Errors or corrections? Email us at aws-course-feedback@amazon.com.
Other questions? Contact us at
https://aws.amazon.com/contact-us/aws-training/.



Module 1: Designing & building the API Backend

Using Amazon API Gateway, AWS Lambda, Amazon DynamoDB and Amazon EC2

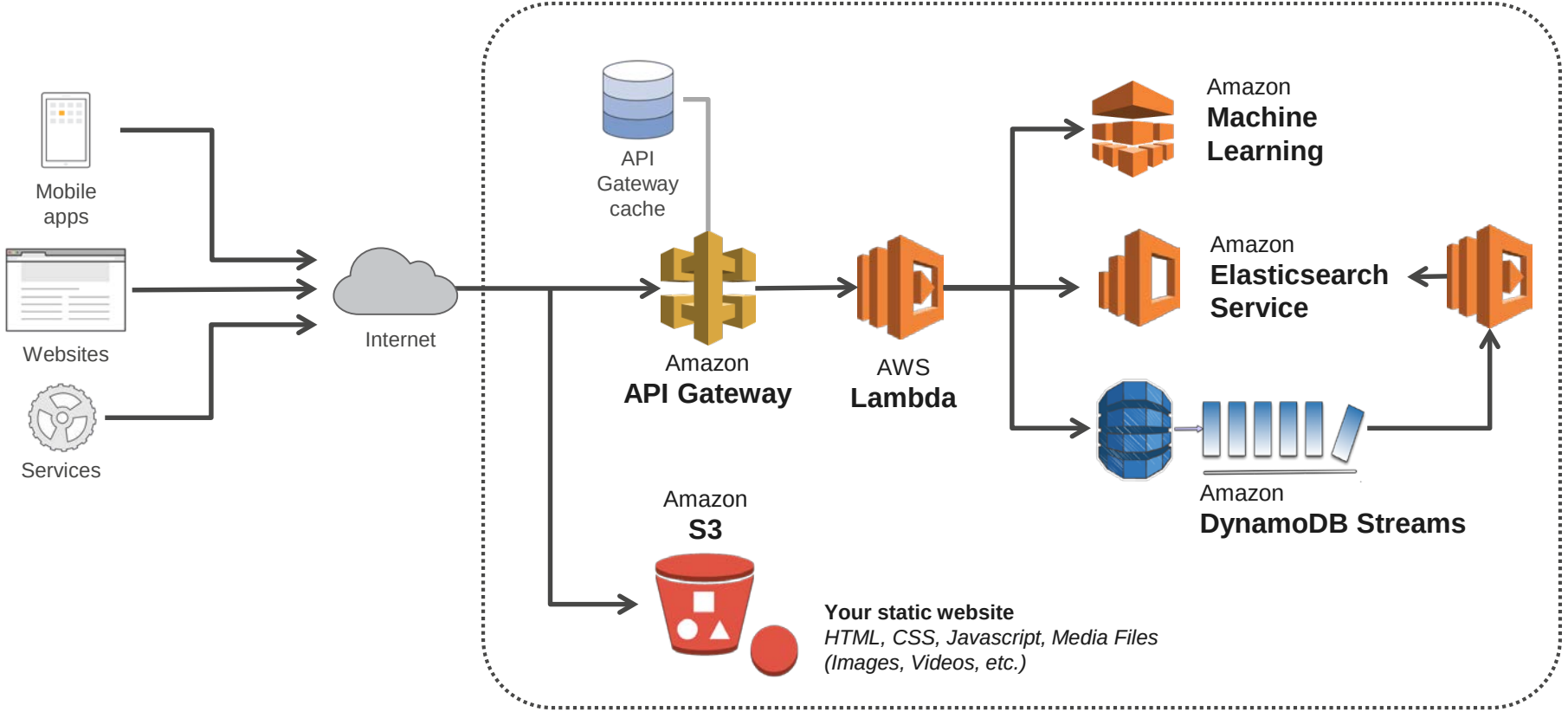
Module Overview

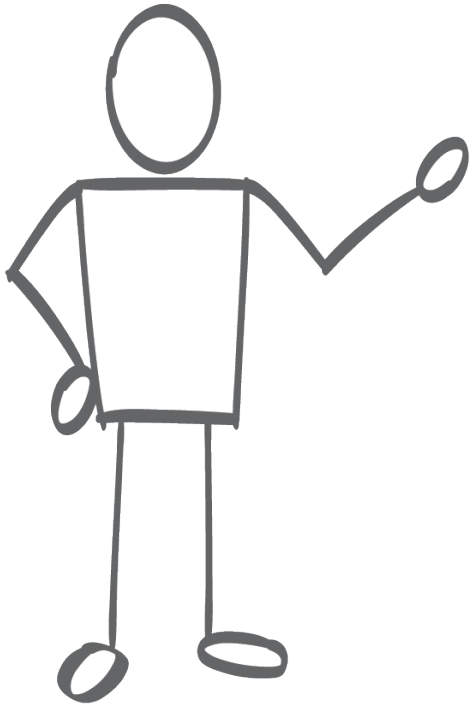
Goal: Build APIs used by the web application.

- Using Amazon API Gateway to define the API
- Creating AWS Lambda functions for custom logic
- Using Amazon DynamoDB for storing retrieving data
- Using DynamoDB Streams to update ElasticSearch
- Performing Geospatial queries using ElasticSearch

In the Lab: Create an API using API Gateway and link it to AWS Lambda functions. These functions will:

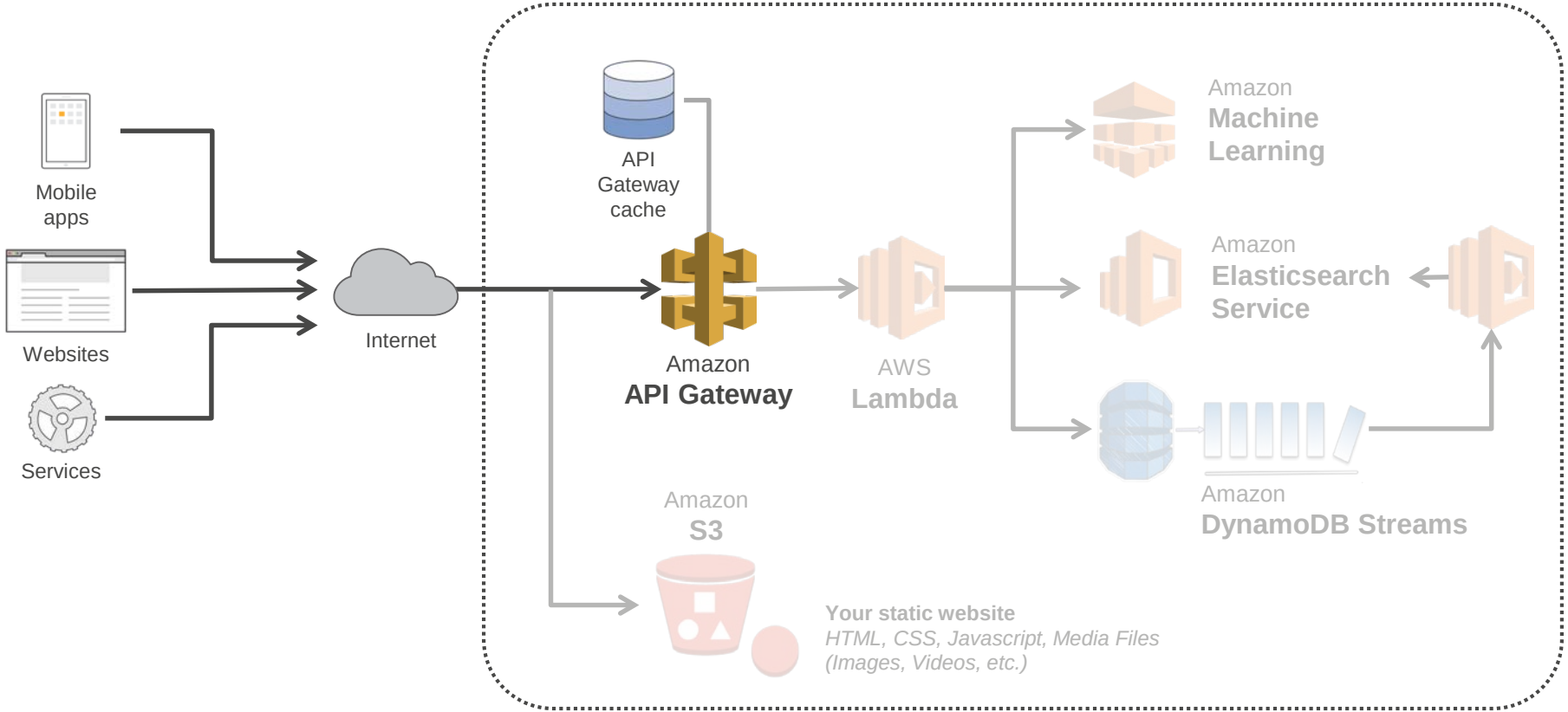
- Query Amazon Elasticsearch Service.
- Read and write from DynamoDB.
- Get results from AWS Machine Learning.





Amazon API Gateway

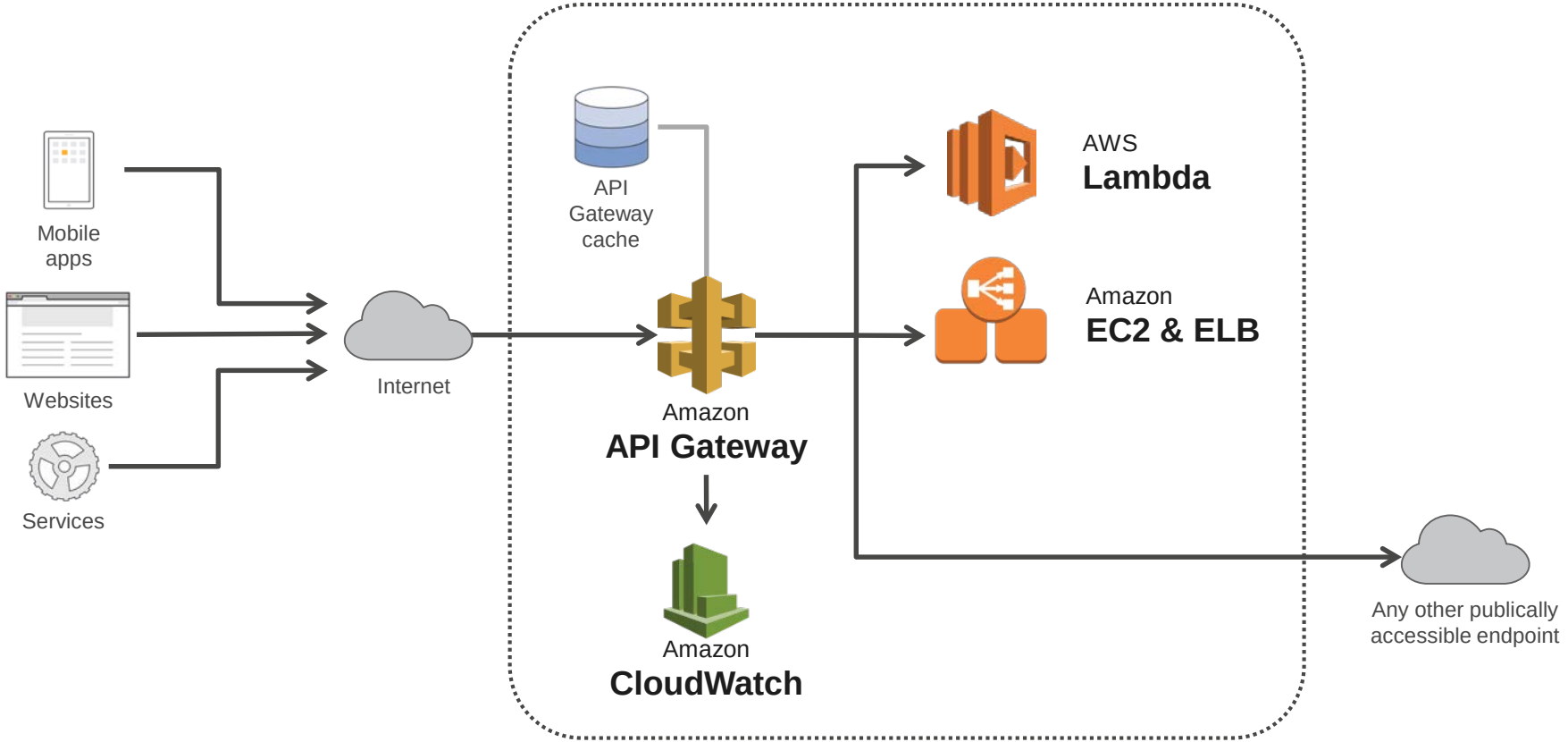
Architecture: Amazon API Gateway



Amazon API Gateway Features

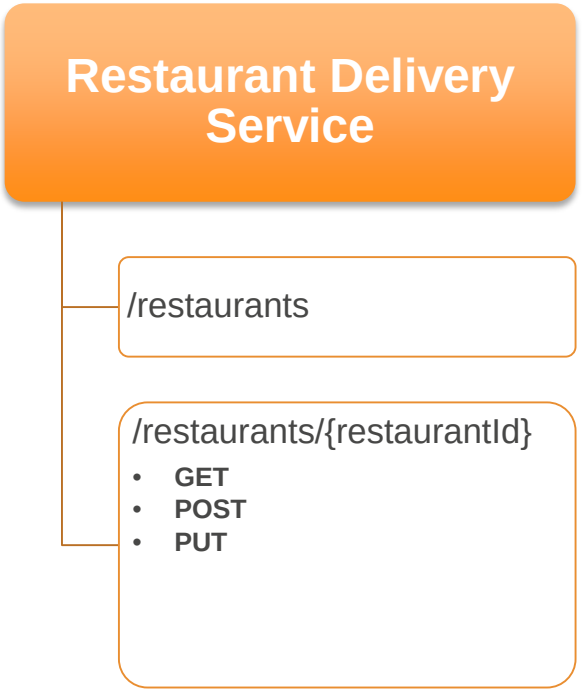
- Hosts multiple versions and stages of APIs
- Creates and distributes API keys to developers
- Leverages signature version 4 to authorize access to APIs
- Throttles and monitors requests to protect the backend
- Uses AWS Lambda for complete server less implementation
- Utilizes a managed cache to store API responses
- Allows you to perform request/response data transformation
- SDK generation for iOS, Android and JavaScript

How Amazon API Gateway Works



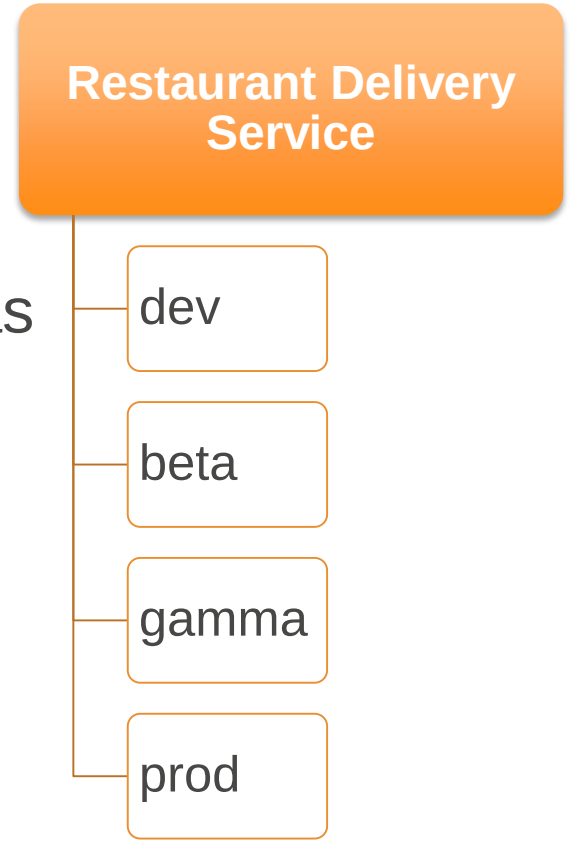
API Configuration

- Top Level REST API
- Define resources within as API
- Define methods for a resource
  - Methods are resource + HTTP verb
  - API Gateway supports 7 standard HTTP verbs

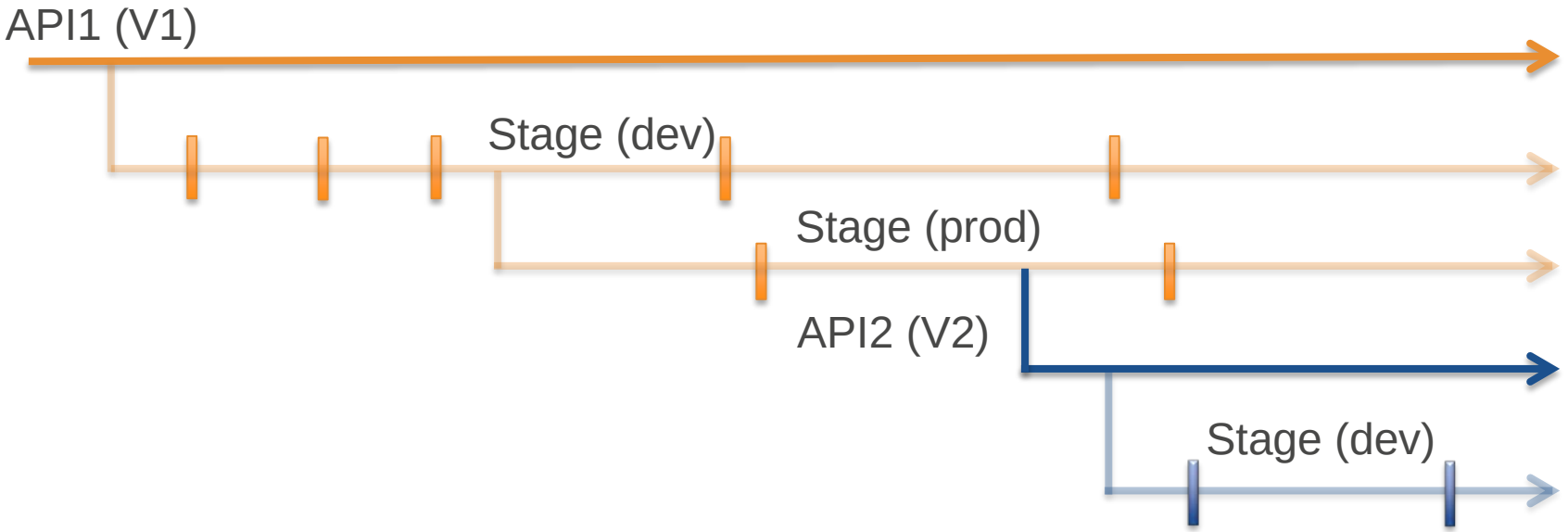


API Deployments

- API configuration can be deployed to a stage
- Stages are different environments such as
  - Dev
  - Beta
  - Prod
  - As many stages as you need



Amazon API Gateway



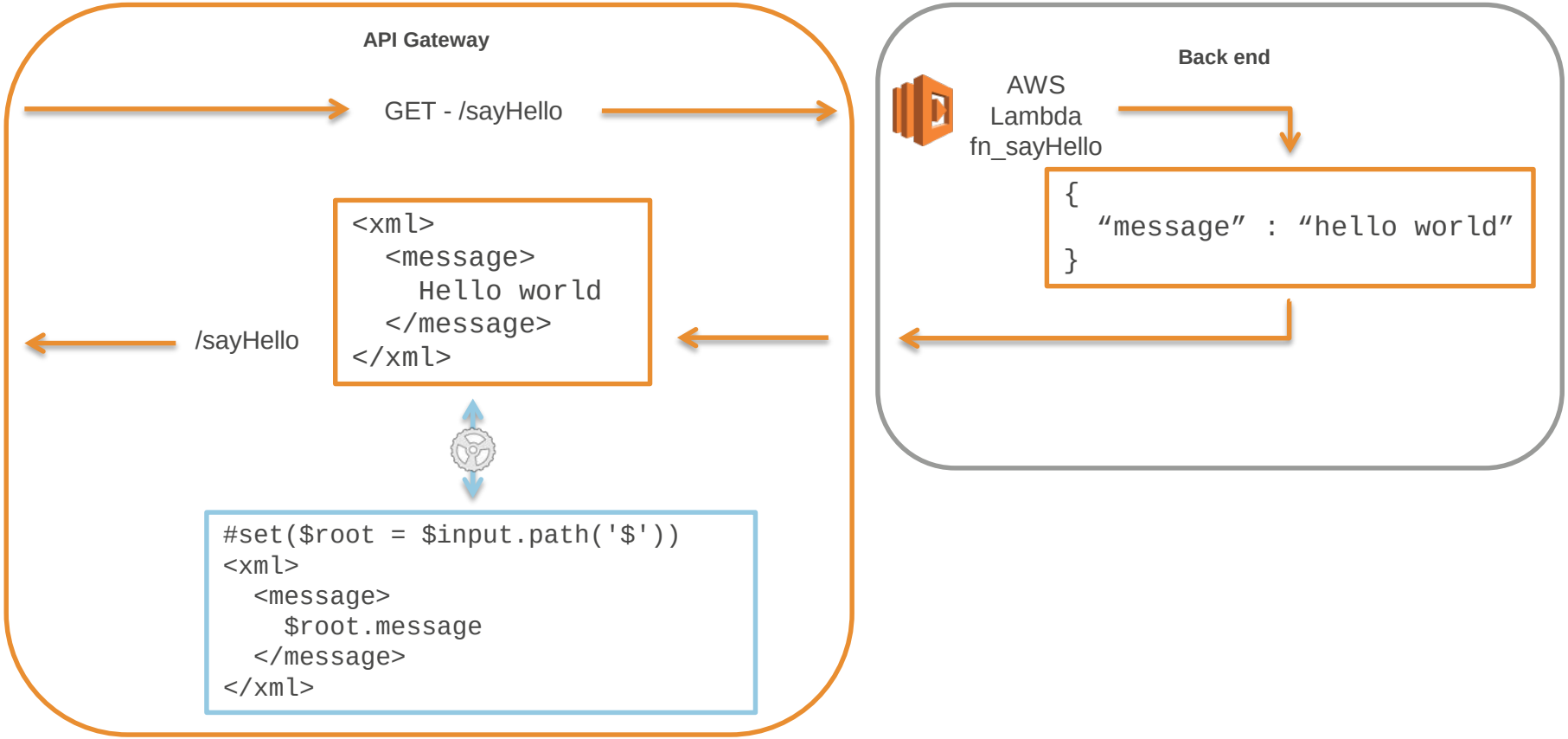
Caching API Responses

- You can configure a cache key and Time to Live (TTL) of the API response
- Cached items are returned without calling the backend
- A cache is dedicated by stage

Input/Output Models

- Models are a JSON schema representation of your API requests and responses
- Models are used for input and output filtering and SDK generation
- You can reuse models across multiple methods in your API

Input/Output Transforms

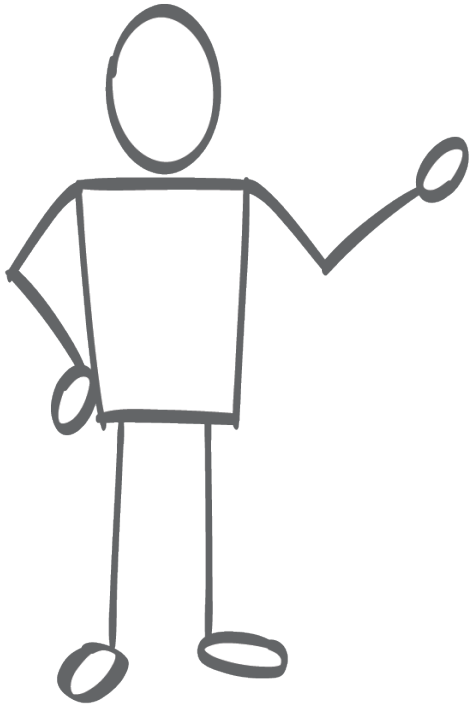


Other Features of Amazon API Gateway

- Custom domain names
  - You can provide API Gateway with a signed HTTPS certificate
- API Keys can be used for metering usage
- API Authorization
  - Use AWS signature version 4 to sign and authorize API calls
    - Amazon Cognito or AWS Security Token Service (STS)
  - Oauth can be set up by forwarding the customer headers from API methods to your back end
- API calls can be throttled to a predefined rate

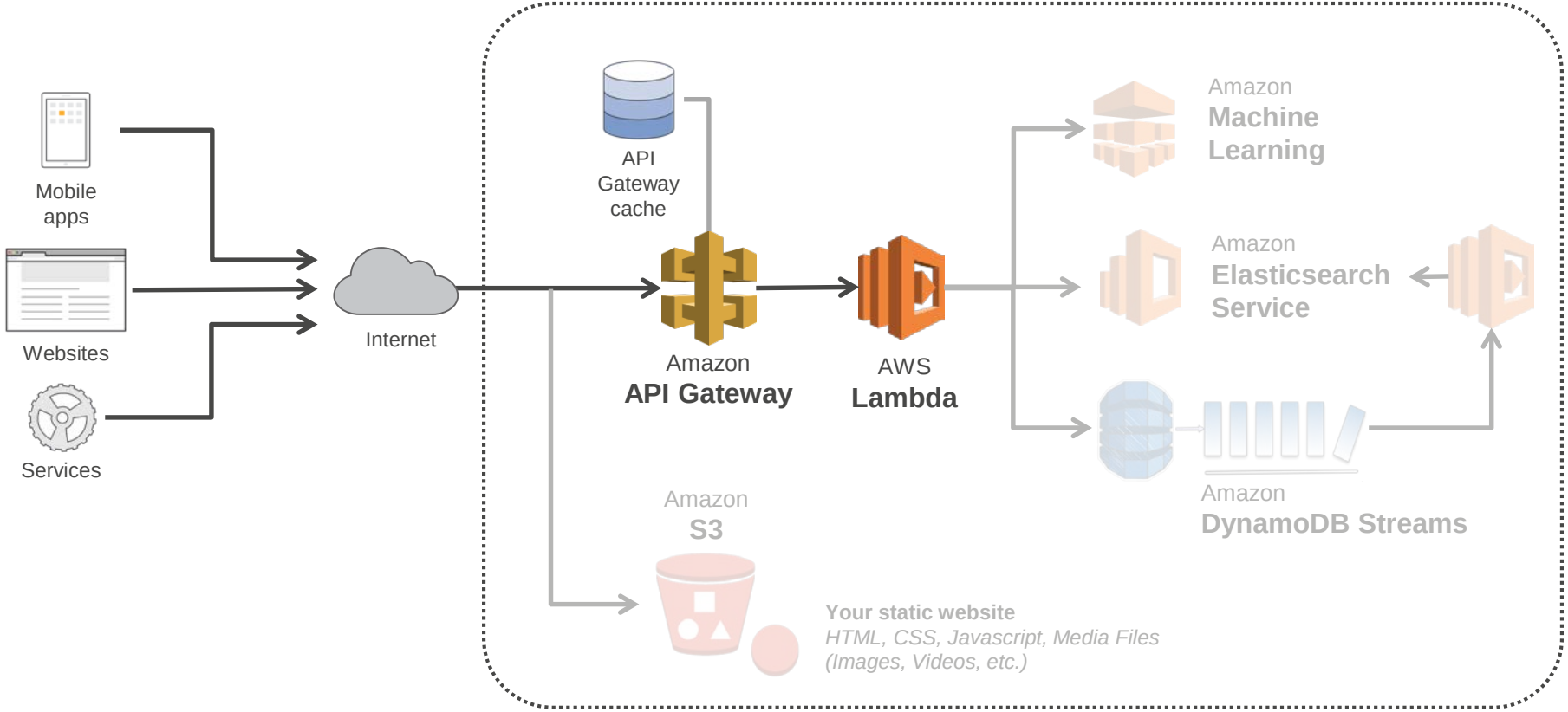
Generate Client SDKs Based on APIs

- SDKs are generated based on API deployments
- If request-response models are defined, the SDK includes input and output marshaling of methods
- SDKs know how to handle throttling responses
- SDKs know how to sign requests with AWS temporary credentials (signature version 4)
- Support for Android, iOS, JavaScript



Using AWS Lambda

Architecture: AWS Lambda

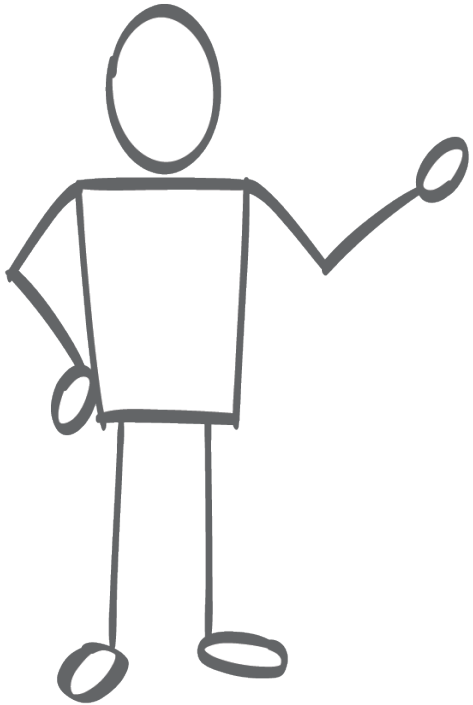


AWS Lambda Features

- No Infrastructure to Manage
- Automatic scaling
- Bring your own code
- Integrated with many AWS services
- Pay for what you use; billing per 100ms increment
- Automatically integrated with Amazon CloudWatch logs

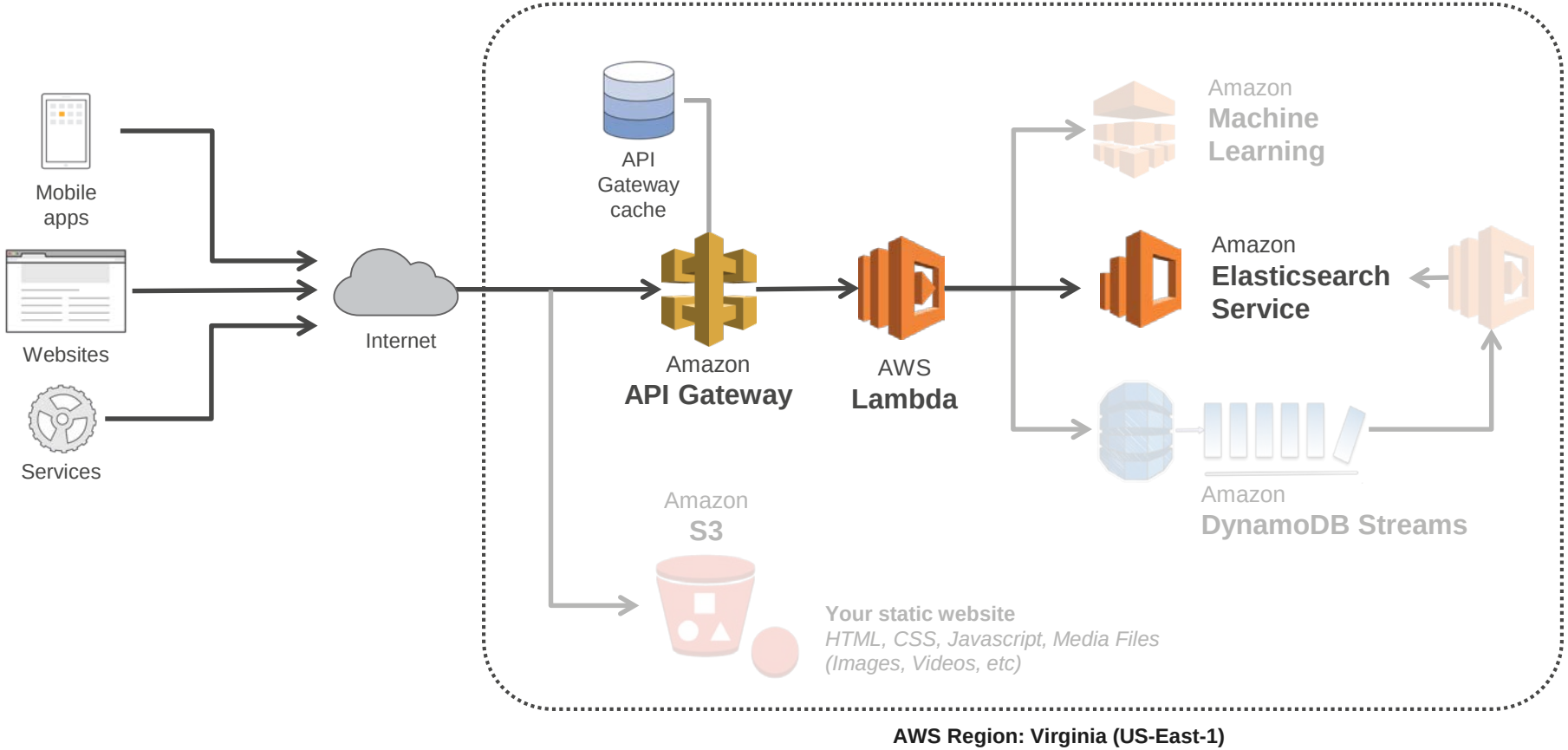
Writing Lambda Functions

- Pick a name
- Pick a size
  - Memory from 128MB to 1.5 GB, in 64 MB steps
  - Receive an equivalent amount of other resources (disc, network, compute, etc.)
- Pick an event source or call sync (we'll use API Gateway)
- More in the Lab…



Using Amazon Elasticsearch Service

Architecture: Elasticsearch



Elasticsearch - A search engine for analytics?

- Simplicity – Schema free, Elasticsearch/Logstash/Kibana (ELK), open source
- Flexibility – A search engine
- Efficiency – Queries are fast
- Query power – Ad hoc statistics and aggregations
- Kibana – Powerful UI

Elasticsearch Setup in Today's Lab

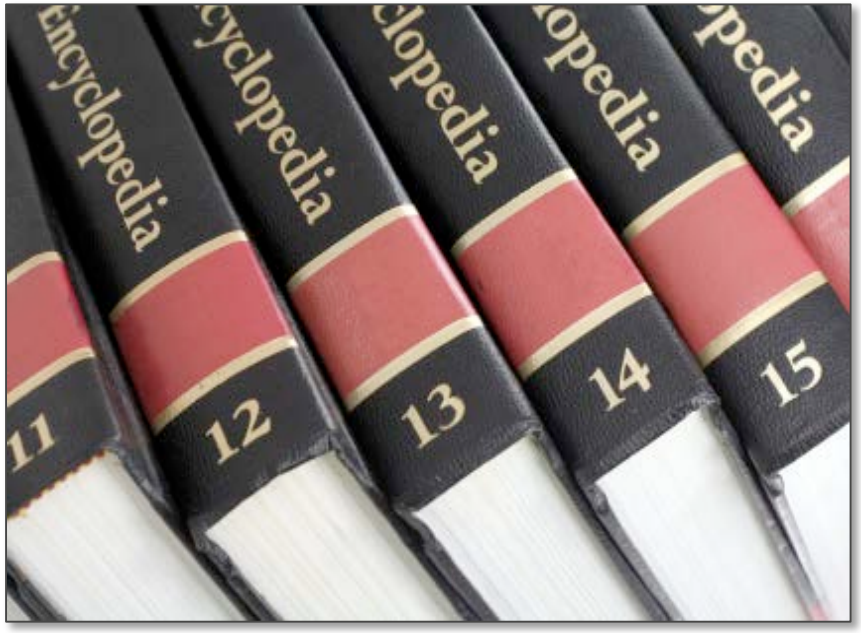
- AWS CloudFormation template creates a single node Amazon Elasticsearch Service cluster
- Elasticsearch daemon is run using bootstrap actions
- Elasticsearch data is exposed to the user interface through Amazon API Gateway

Elasticsearch Interactions

- Prepare index documents; you can use Amazon DynamoDB Streams consumer AWS Lambda function
- Retrieve documents that match query terms – API gateway transforms the request and responses to the form required by the client and the Lambda functions.
- Sort and collate responses – Lambda function combines the results from Elasticsearch with Amazon Machine Learning to produce the desired results

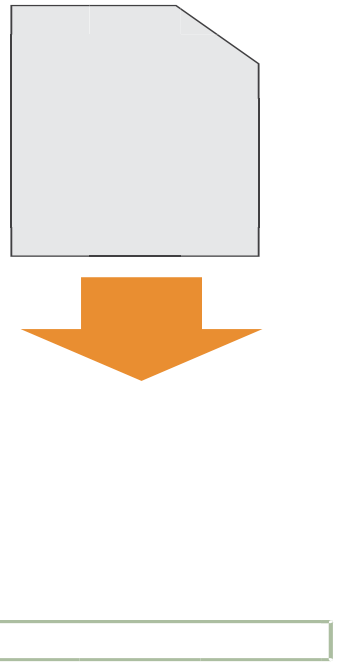
Search Engine Structure

- Search engines hold documents
- Documents contain fields
- Fields contain one or many values



Indexing

- Inverted index maps words to documents
- Each word is an entry in the index
- Query processing is O(1) lookup plus merging and scoring



Scoring

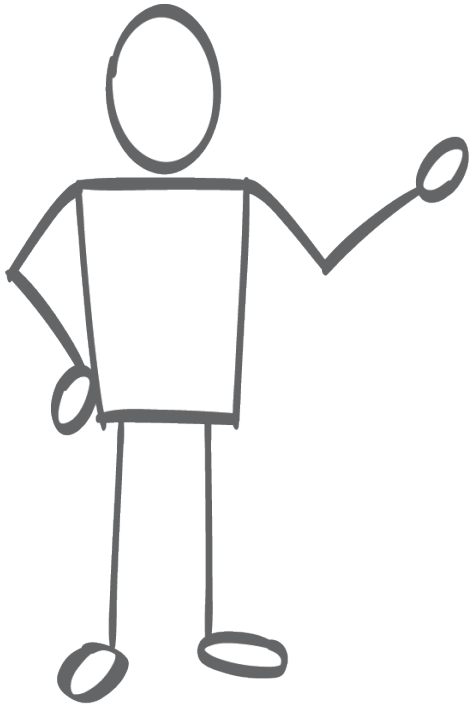
- Relevance differentiates search and database
  - Tf = Term Frequency
  - Df = Document Frequency
- Per document score is the normalized product of tf * 1/df

Location-Based Search Using Elasticsearch

- Geopoint - A single latitude/longitude point on the Earth's surface
- Geohashes - A way of encoding lat/lon points as strings. Geohashes divide the world into a grid of 32 cells—4 rows and 8 columns—each represented by a letter or number.

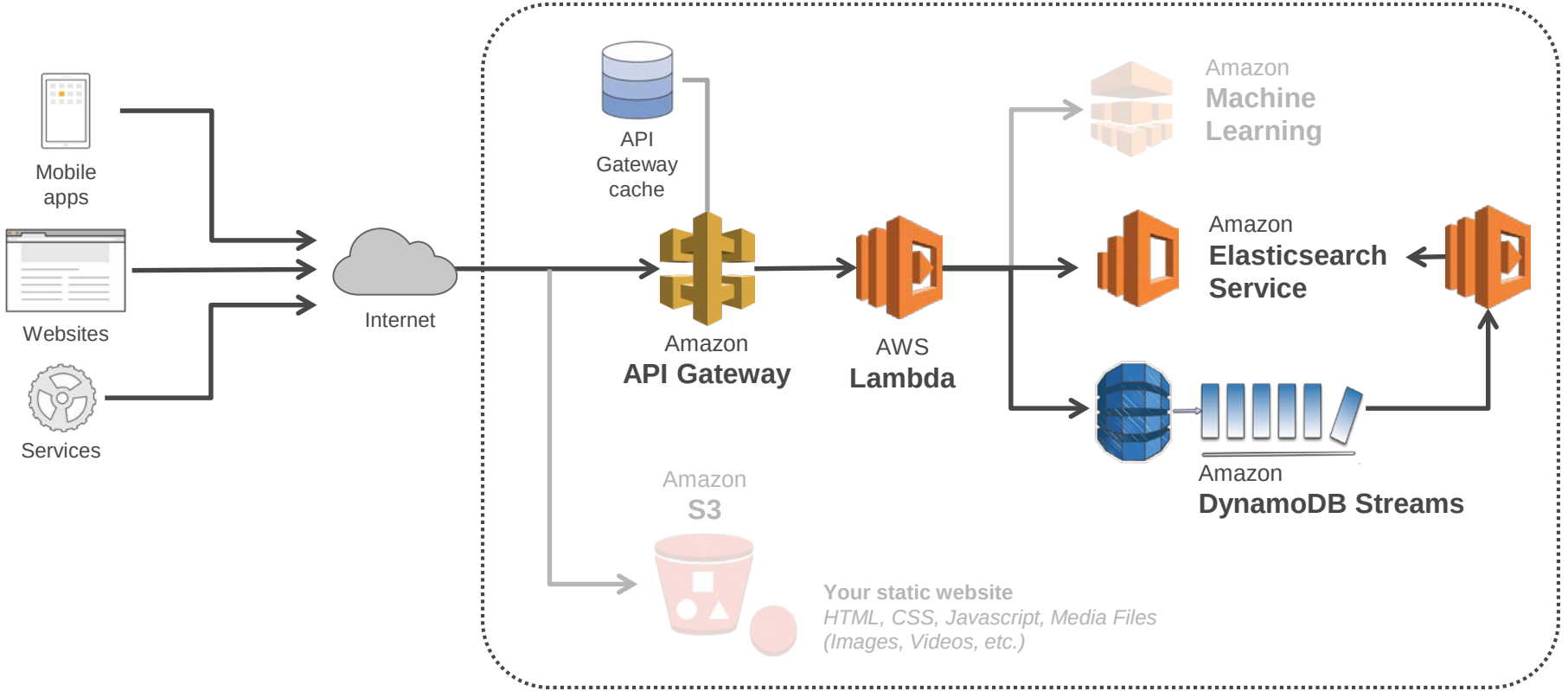
Location-Based Search Example

```
GET /restaurant/_search
{
  "query": {
    "filtered": {
      "filter": {
        "geohash_cell": {
          "location": {
            "lat":  40.718,
            "lon": -73.983
          },
          "neighbors": true,
          "precision": "2km"
        }
```



Using Amazon DynamoDB Streams

Architecture: Amazon DynamoDB Streams



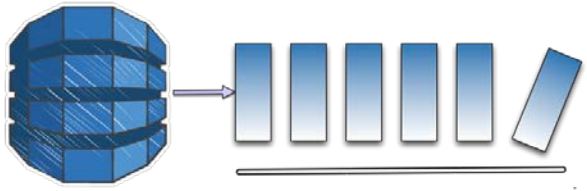
Amazon DynamoDB Features

- Is a managed NoSQL database service
- Supports both document and key-value data models
- Is highly scalable
- Provides consistent, single-digit millisecond latency at any scale
- Is highly available – 3x replication
- Has a simple and powerful API

Amazon DynamoDB Streams

- Stream of updates to a table
- Asynchronous
- Exactly once
- Strictly ordered (per item)
- Highly durable
- Scale with table
- 24-hour lifetime
- Sub-second latency

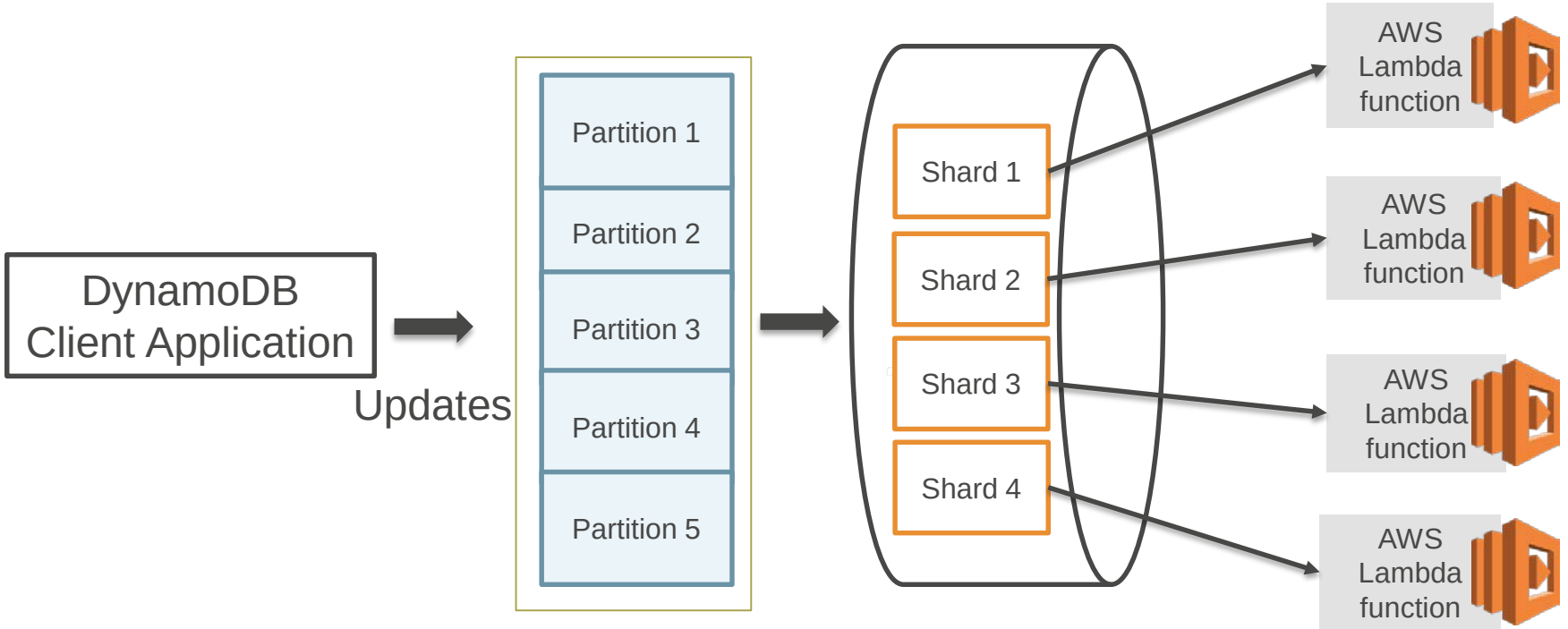
View Types



UpdateItem (Name = John, Destination = Pluto)

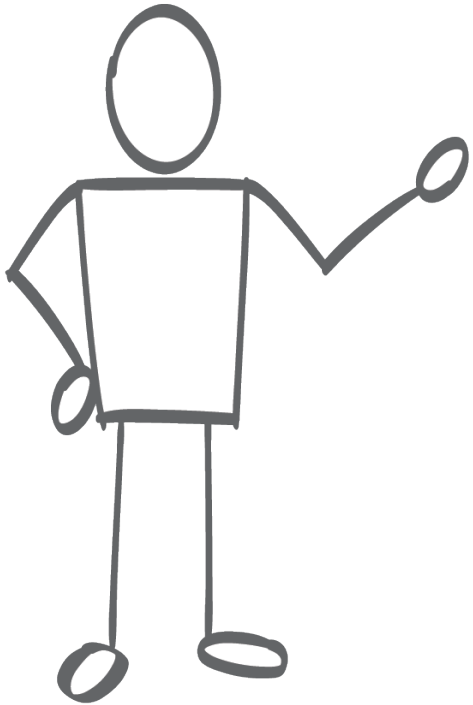
View Type	Destination
Old image—before update	Name = John, Destination = Mars
New image—after update	Name = John, Destination = Pluto
Old and new images	Name = John, Destination = Mars Name = John, Destination = Pluto
Keys only	Name = John

Consuming Streams Data



Lab : Creating the APIs

Errors or corrections? Email us at aws-course-feedback@amazon.com.
Other questions? Contact us at
https://aws.amazon.com/contact-us/aws-training/.



Module 2: Building a Serverless Web Application

Using Amazon S3, Mapbox, Yeoman and Amazon API Gateway

Module Overview

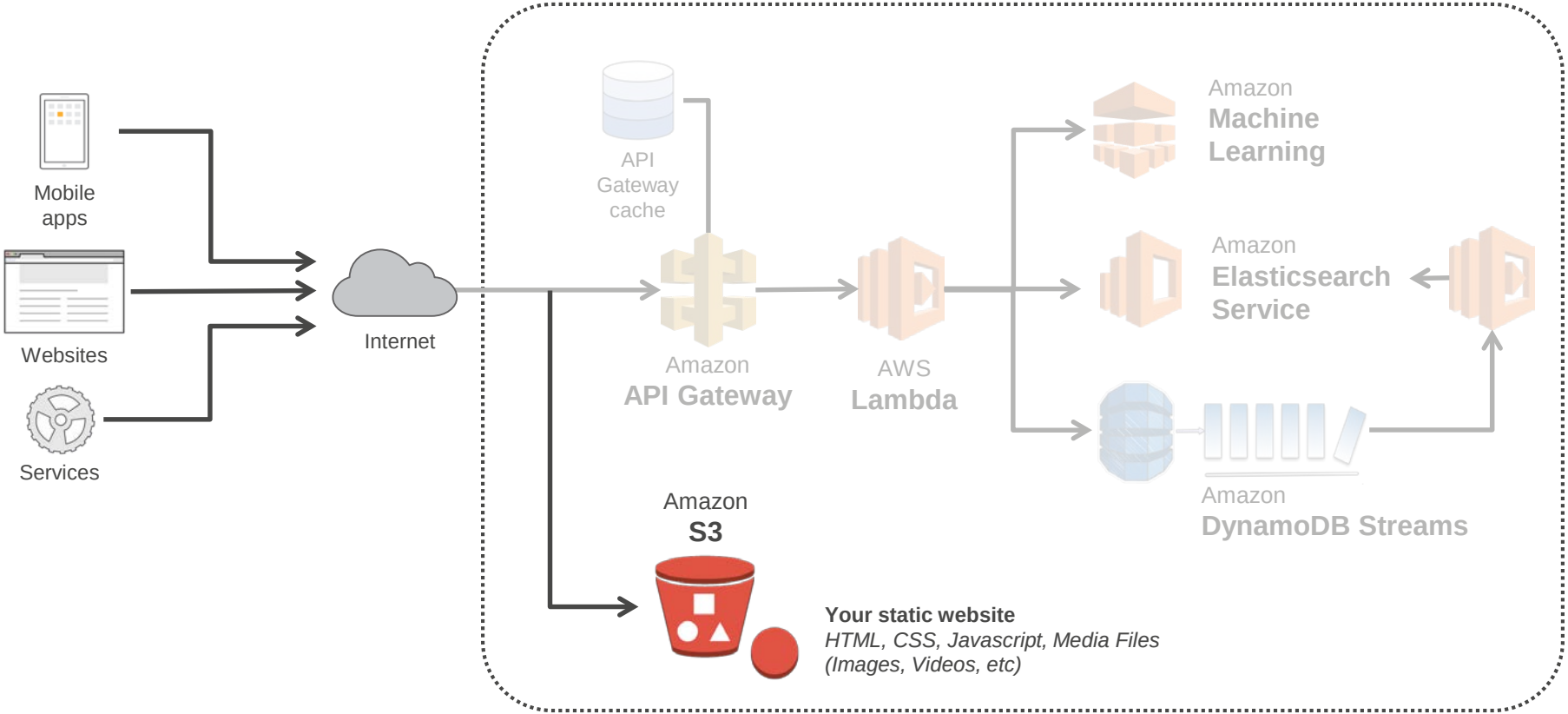
Goal: To learn about serverless computing.

- Running highly available and scalable architectures without managing EC2 instances.
- Paying only for what you use, enabling you to match your revenue model with your projected spend.

In the Lab: Develop and deploy a web application to Amazon S3 to visualize your geospatial data.

- Learn how to use scaffolding/development tools.
- Learn how to create an S3 static website.
- Publish your web application code to S3.

Architecture: Amazon S3 Website



What is Serverless Architecture?

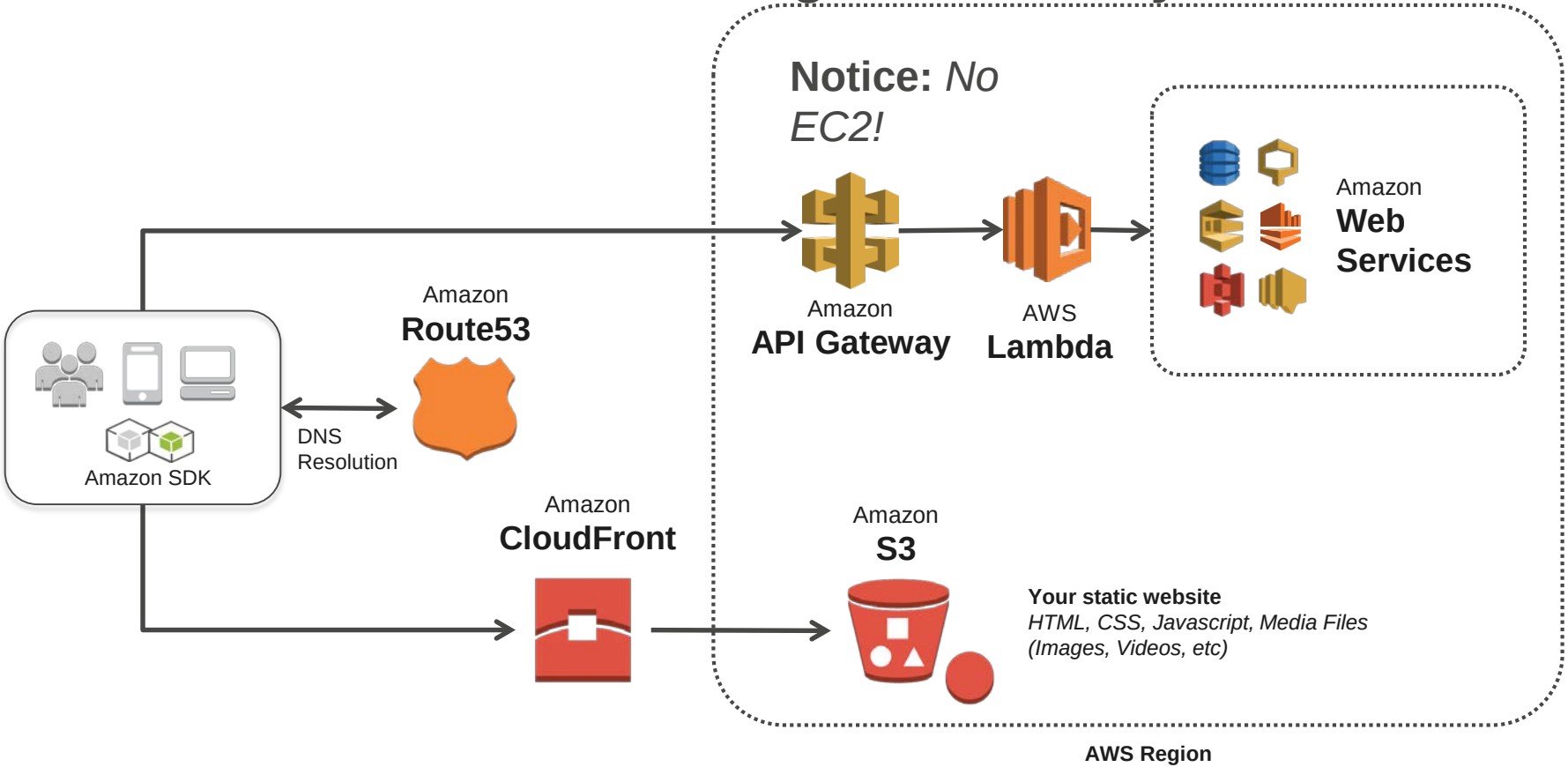
Of course, there are *servers* involved. You just don't have to manage any of them.

You don't have to manage **patching**, **sizing, scaling**, **performance tweaking** etc.

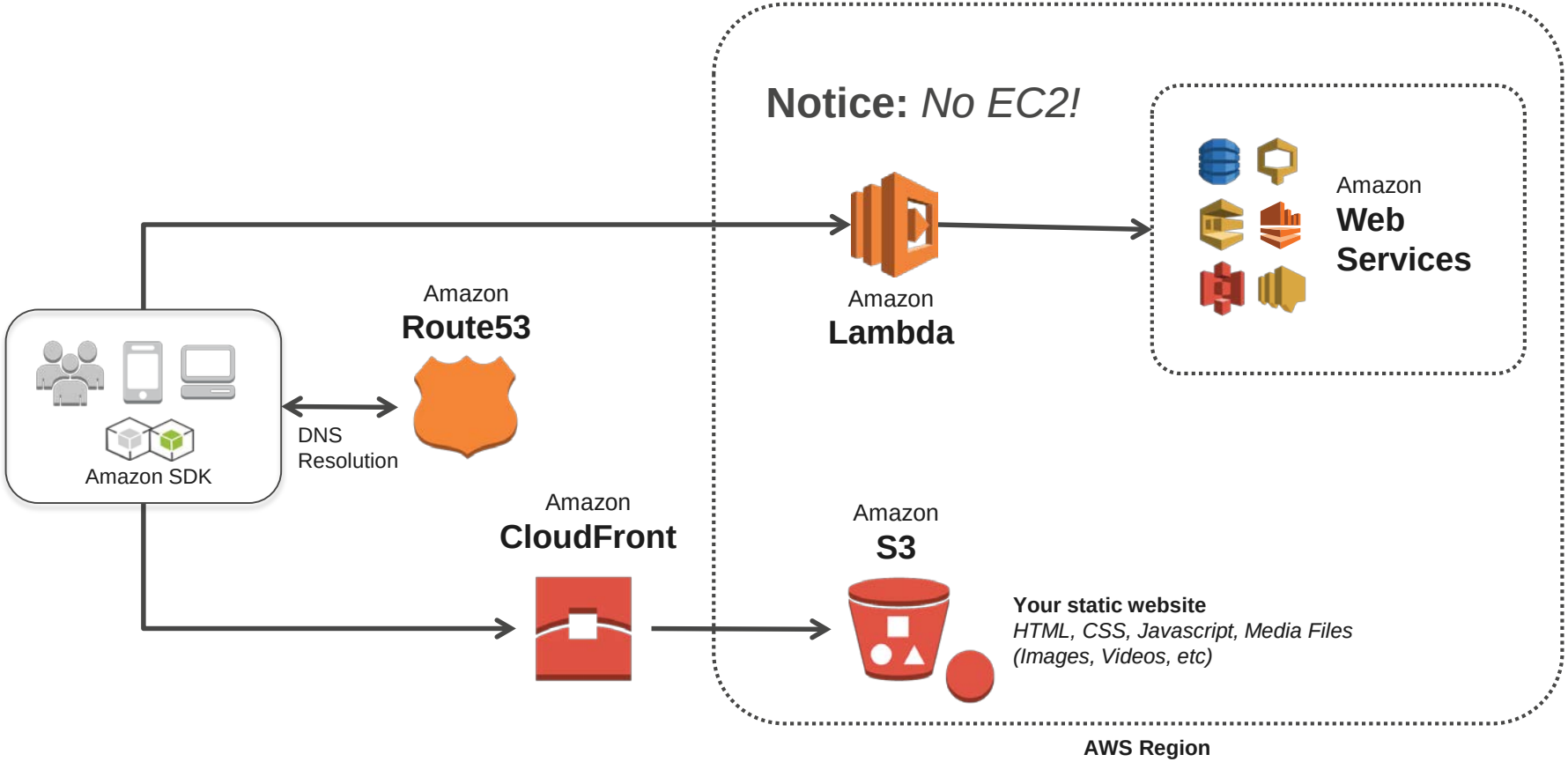
The application files (HTML, CSS, JS, Media files etc) are downloaded and rendered in the user's browser.

Dynamic business logic is encapsulated and run in the cloud by AWS Lambda functions.

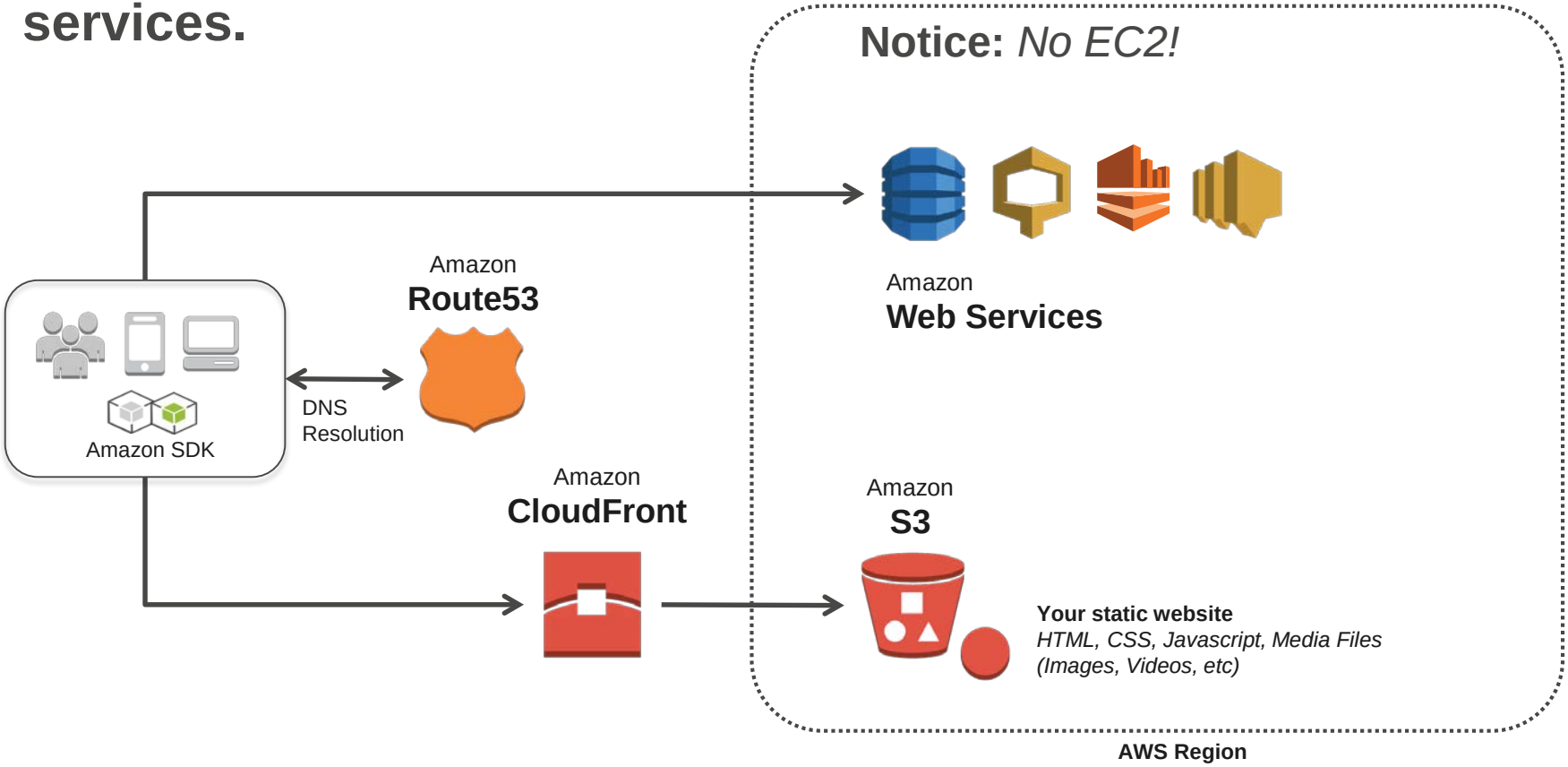
Serverless Architecture Using API Gateway



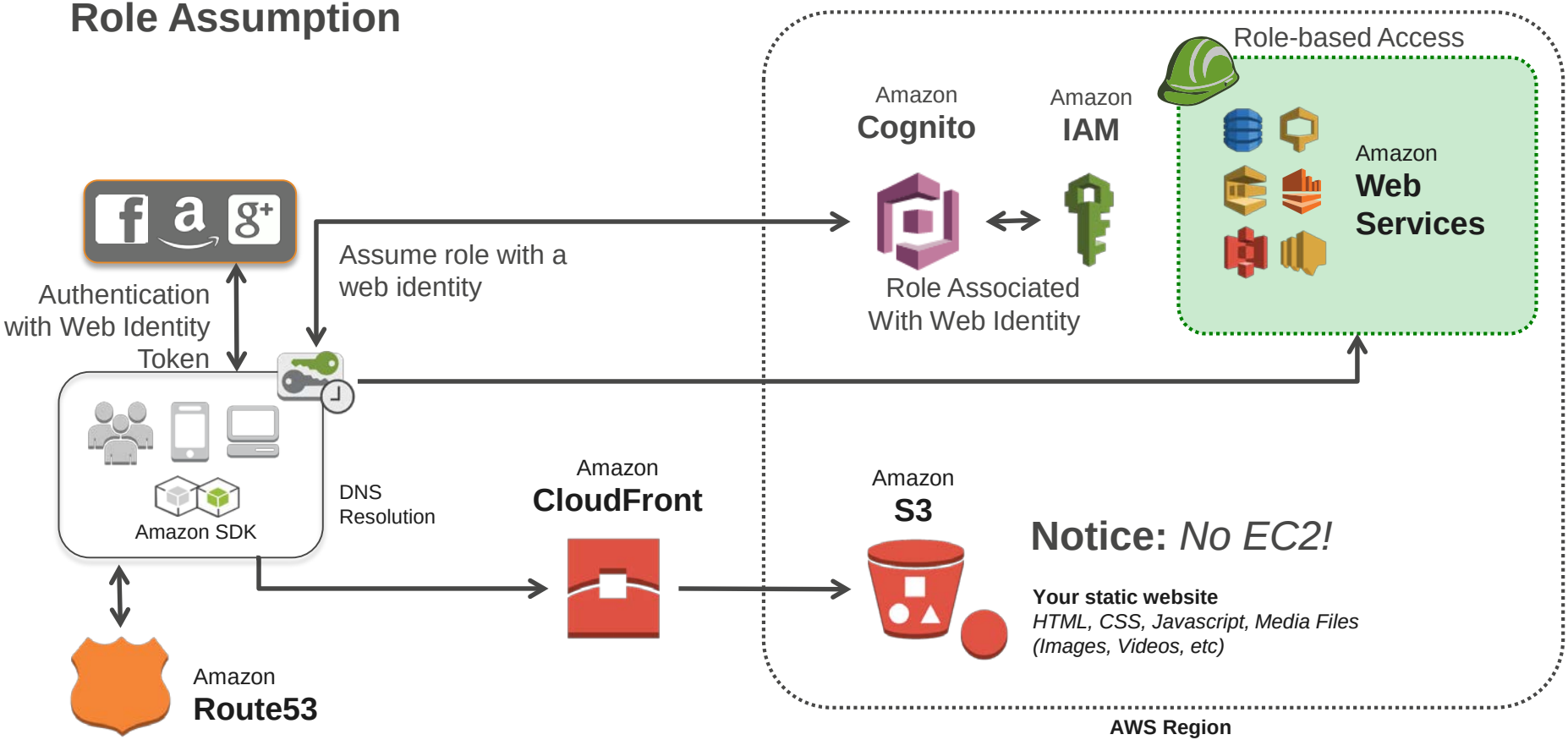
Your applications *could* invoke Lambda functions directly.



Or, they *could* communicate directly with a number of services.

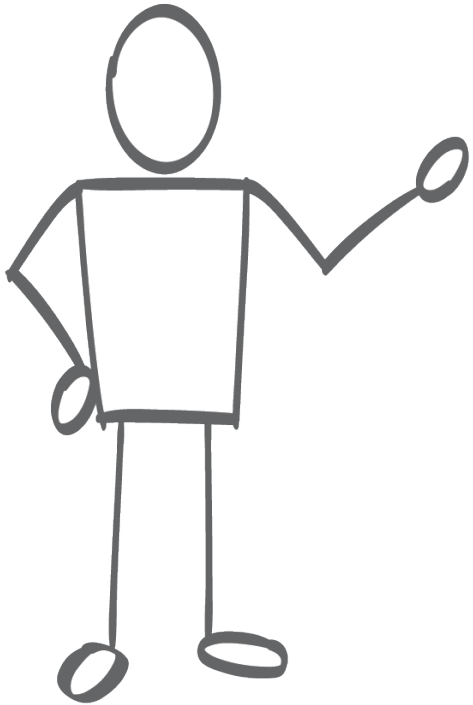


For security, you *could* use Amazon Cognito for Identity Federation and Role Assumption



To Build the Serverless Web Application:

- **Responsive** web application using **Twitter Bootstrap** and delivered by **Amazon S3**.
- **Mapbox** for mapping.
- Elasticsearch on EC2, Amazon DynamoDB and Amazon Machine Learning accessed via RESTful API requests to **Amazon API Gateway** and **AWS Lambda**.
- Node.JS scaffolding tool for dev/test.
  - For this lab, you will use Yeoman.



The Development Environment

Using Yeoman

What is a static site generator?

- Acts as scaffolding tools for web applications.
- Enables you to get a skeleton web application up and running quickly.
- There are many tools out there that do a similar thing. Yours will be built based on **Yeoman** because it is has wide adoption and good documentation.

- Has a wide range of **generators** that build out the scaffolding required for many types of web applications, including:
  - Bootstrap
  - Angular.JS
  - Ionic
  - Wordpress
  - React.JS
  - ASP.NET apps
  - Django
  - And more!

Requirements for the Site Generator

To get started with our static site generator, you will need to have the following dependencies installed:

- **Node.JS**: A Javascript platform.
  - Uses an event-driven, non-blocking I/O model.
  - Lightweight, perfect for data-intensive real-time applications.
- **Bower**: A package manager designed for front-end packages.
- **Grunt**: A task runner written in Javascript.
  - Makes running repetitive tasks easier.

Developing Serverless Web Applications

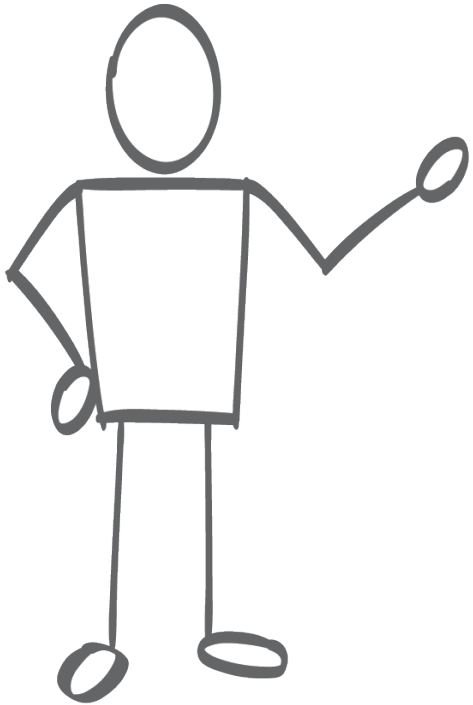
- Open the folder with your favorite text editor. A couple good ones are **Sublime Text 2, Atom** and **Visual Studio Code** .
- Open a terminal window beside your code editor and change the directory to point to your code folder.
- Run **grunt serve**. Grunt will start a local webserver and is watching for changes in your code. If a change is detected, it will automatically re-render the website in your browser.
- Now, make a small change to the HTML of the website. You should see the changes reflected in your open browser window.
- Demo…

Testing with Grunt

- Yeoman uses the Javascript test runner **Karma.**
- To run your tests, simply type **grunt test.**
- Your unit tests can be stored in the **test/** directory.

Publishing to Amazon S3 with Grunt

- A simple grunt task will publish to your S3 bucket.
- Update **aws.json** with your Amazon S3 bucket name and console profile name (if you have one, if not, leave as default)
- To publish, simply run **grunt publish.**



The Web Application

Using Amazon S3, Mapbox and API Gateway

Displaying Geo-data with Mapbox

- Mapbox is a powerful mapping service that makes it easy to integrate maps into your web and mobile applications.
- You will display your geo-data using **Mapbox**.

An Overview of Static Websites on S3.

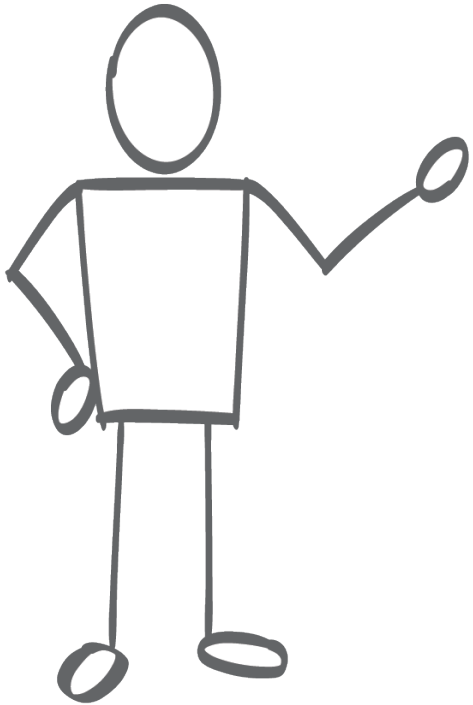
- Static content (HTML,CSS, JS, Media files, etc.) can be served by S3 directly.
- By contrast, a dynamic website relies on server-side processing, including server-side scripts such as PHP, JSP, or ASP.NET. Amazon S3 **does not** support server-side scripting.
- You will use **AWS Lambda** to process the requests via an **Amazon API Gateway** URL, instead of a backend EC2 instance.

Setting Up Your Static S3 Website.

- In a real-world scenario, you would not use the S3 URL directly. For example:
  - http://**[BUCKET-NAME]**.s3-us-east-1.amazonaws.com/
- Instead, you would point your domain name to the S3 bucket using a DNS service such as **Amazon Route53**.
- It would also be smart to use a content delivery network such as **Amazon CloudFront**, to push the static files to our robust edge network with over 50 points of presence.
- In today's lab, you will reference the S3 URL directly. However, if you have your own domain, feel free to wire it up!

Lab : Building the Web Application

Errors or corrections? Email us at aws-course-feedback@amazon.com.
Other questions? Contact us at
https://aws.amazon.com/contact-us/aws-training/.



Module 3: Introduction to Data Science

Using Amazon Machine Learning

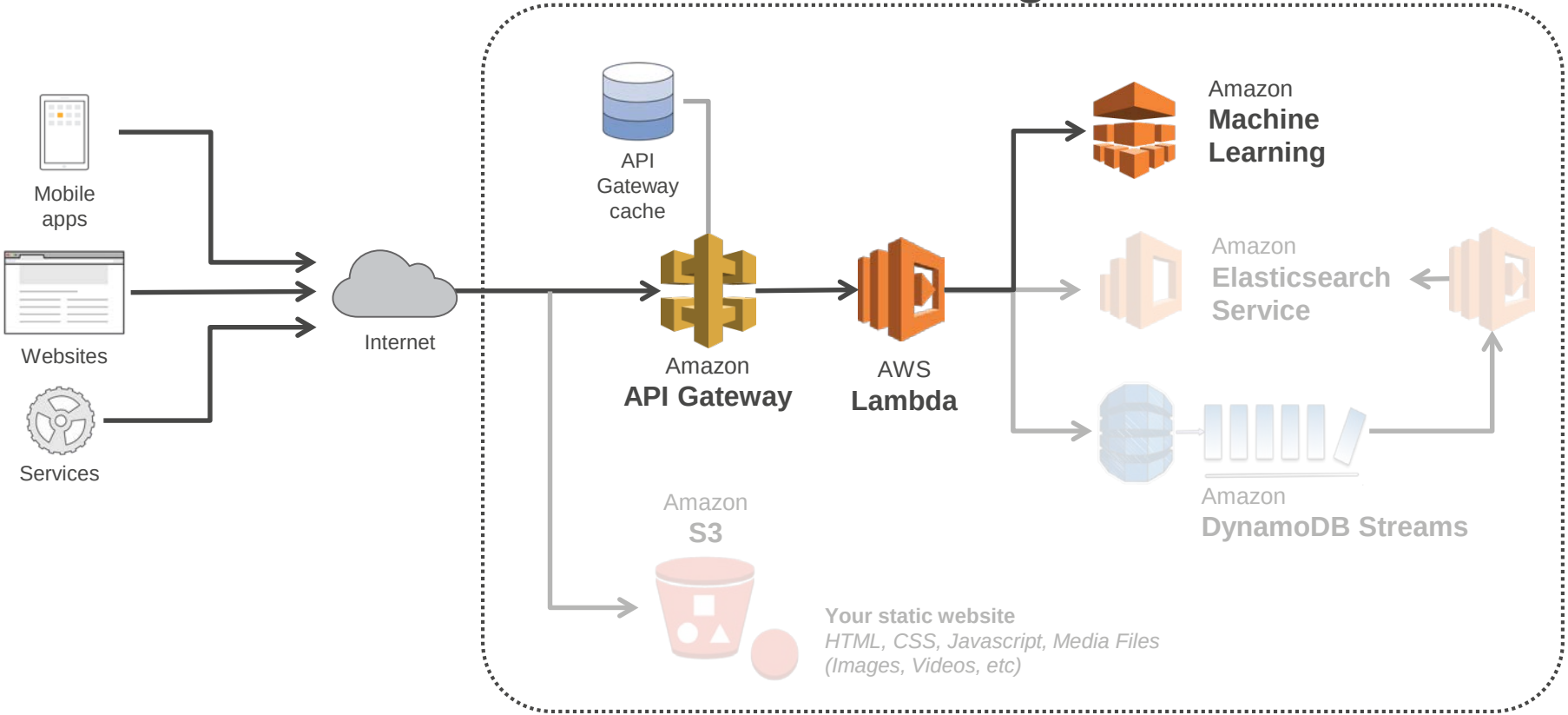
Module Overview

Goal: To better understand the fundamentals of machine learning and to apply a basic machine learning model to your geospatial dataset.

Lab: You will create and evaluate a machine learning model based on an example dataset.

- With the model, you can generate predictions to provide better recommendations for users.

Architecure: Amazon Machine Learning



Agenda

- Overview of Data Science and Machine Learning
- Data Segmentation
- Tree Induction
- Linear Classification
- Logistic Regression
- Over-Fitting
- Conclusion

What is Data Science?

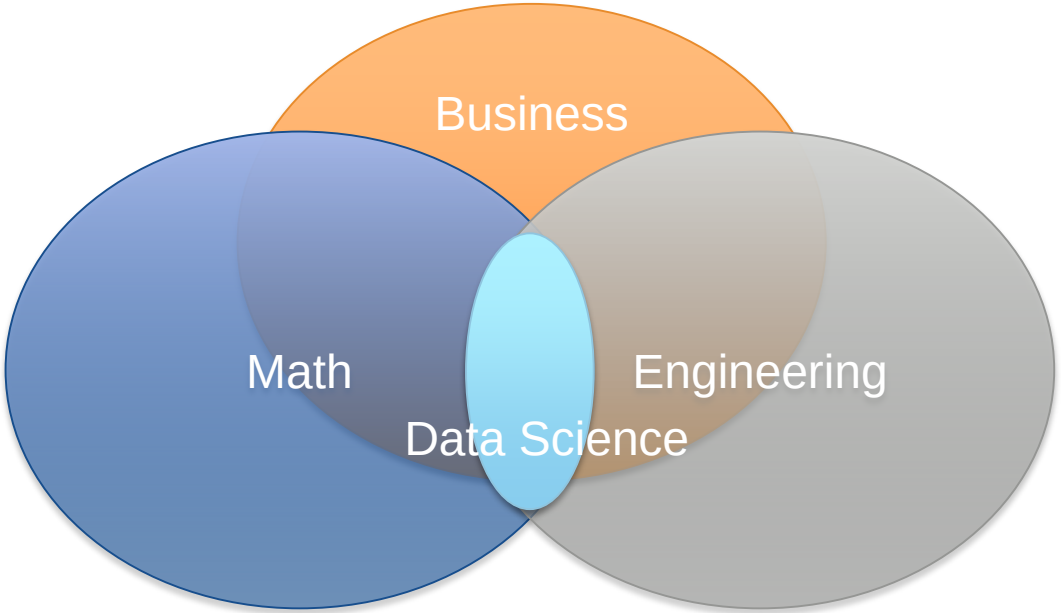
Question: What type of discipline is Data Science?

- Engineering discipline
- Business discipline
- Mathematics discipline

Answer: All Three.

What is Data Science?

Data science is a craft that draws on skills in engineering, math, and business to draw useful conclusions from large data sets.



The Tools of the Data Scientist

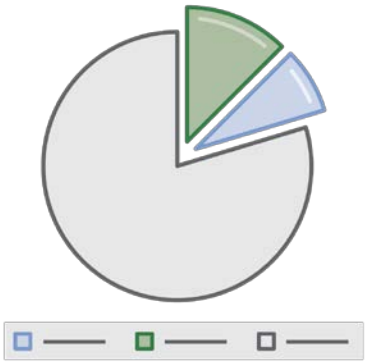
Databases

- SQL Queries
- Used to fetch a specific subset of the data

The Tools of the Data Scientist (Cont'd.)

Summary Statistics

- Average, count, median, standard deviation
- Used to distill large data sets into smaller ones



The Tools of the Data Scientist (Cont'd.)

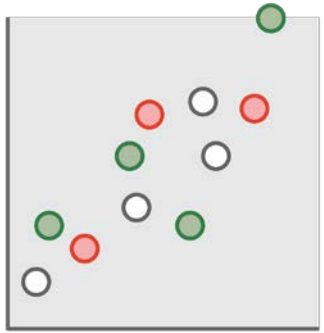
Data Warehousing and Business Intelligence

- Aggregate data from different source systems
- Used to compute summary statistics across many dimensions
- "Show me average sales by quarter by sales team by region"

The Tools of the Data Scientist (Cont'd.)

Regression Analysis

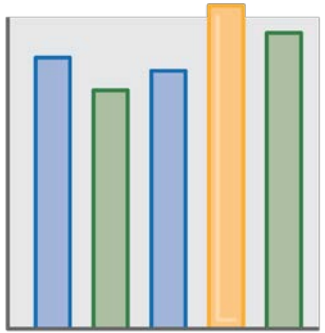
- Used to identify relationships between random variables in a data set



The Tools of the Data Scientist (Cont'd.)

Statistics

- Used to determine if the observed outcomes are real or represent random chance



The Tools of the Data Scientist (Cont'd.)

Machine Learning and Data Mining

- Used to identify patterns in a data set and derive predictions from them

Tool Selection

Pop Quiz – Identify the tool best suited to address the business problem.

Tool Selection

“What were the best selling products from our catalog this month?”

- Machine Learning
- Database Query
- Summary Statistics

Tool Selection

“How many products were sold by category in the U.S regions over the last two quarters?”

- Database Query
- Regression Analysis
- Business Intelligence

Tool Selection

“This report shows that last month, customers with payments overdue by 30 days or more failed to place new orders this month. Is this a real problem for my business?”

- Business Intelligence
- Regression Analysis
- Statistics

Tool Selection

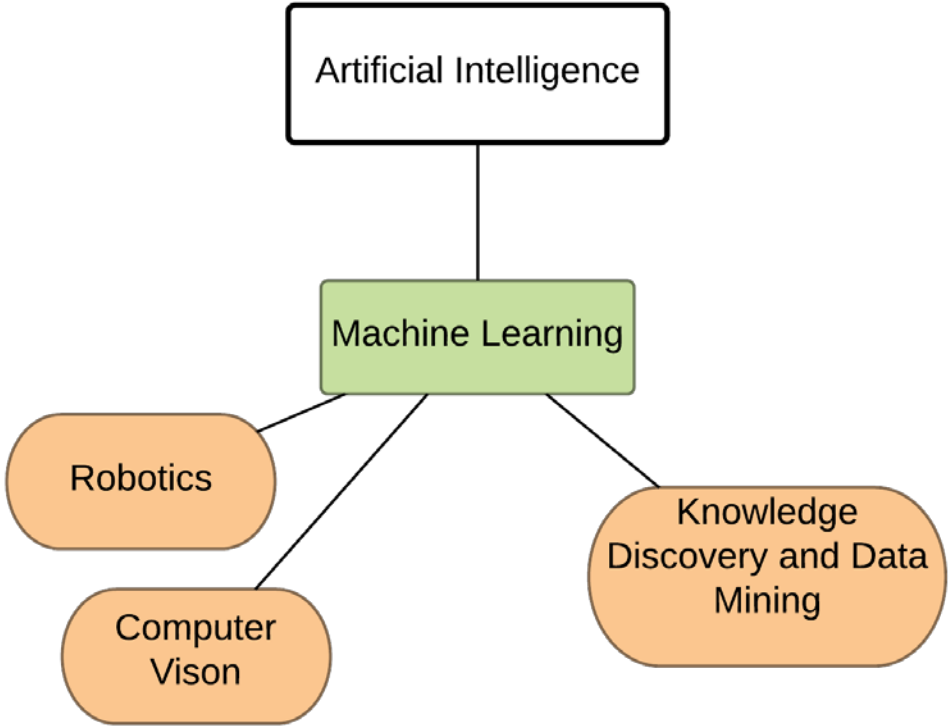
“What customers are most likely to respond favorably to this new marketing campaign?”

- Database query
- Regression Analysis
- Machine Learning

Tool Selection – Review

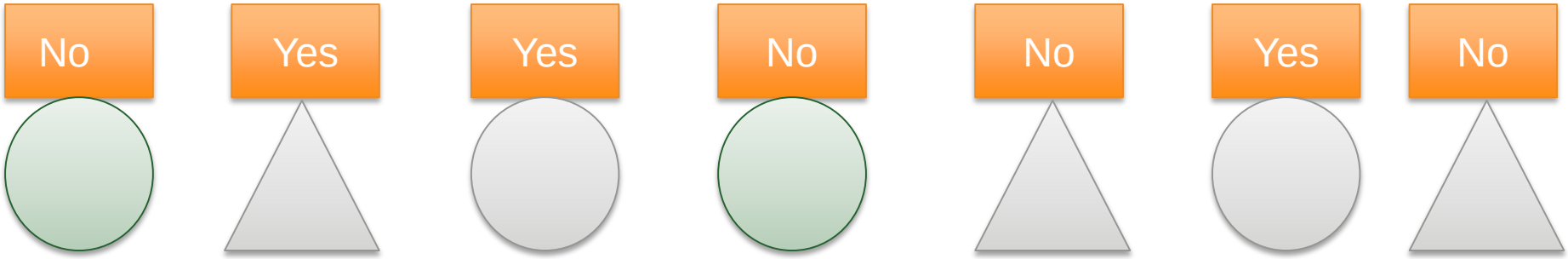
Question	Tool	Selection Criteria
“What were the best selling products this month?”	Database Query	• Information of interest is already known, e.g. “best selling products” • Question is backwards looking
How many products were sold by category in the U.S regions over the last two quarters?	Business Intelligence	• The information of interest spans multiple dimensions • Question is backwards looking
This report shows that last month, customers with payments overdue by 30 days or more failed to place new orders this month. Is this a real problem for my business?	Statistics	• Question is posing a hypothesis • Information of interest is determined by doing a statistical test to compute within a confidence interval whether this is a real event or a random act of chance.
“What customers are most likely to respond favorably to this new marketing campaign?”	Machine Learning	• Information of interest requires making a prediction • Question is forward looking

Origins of Machine Learning

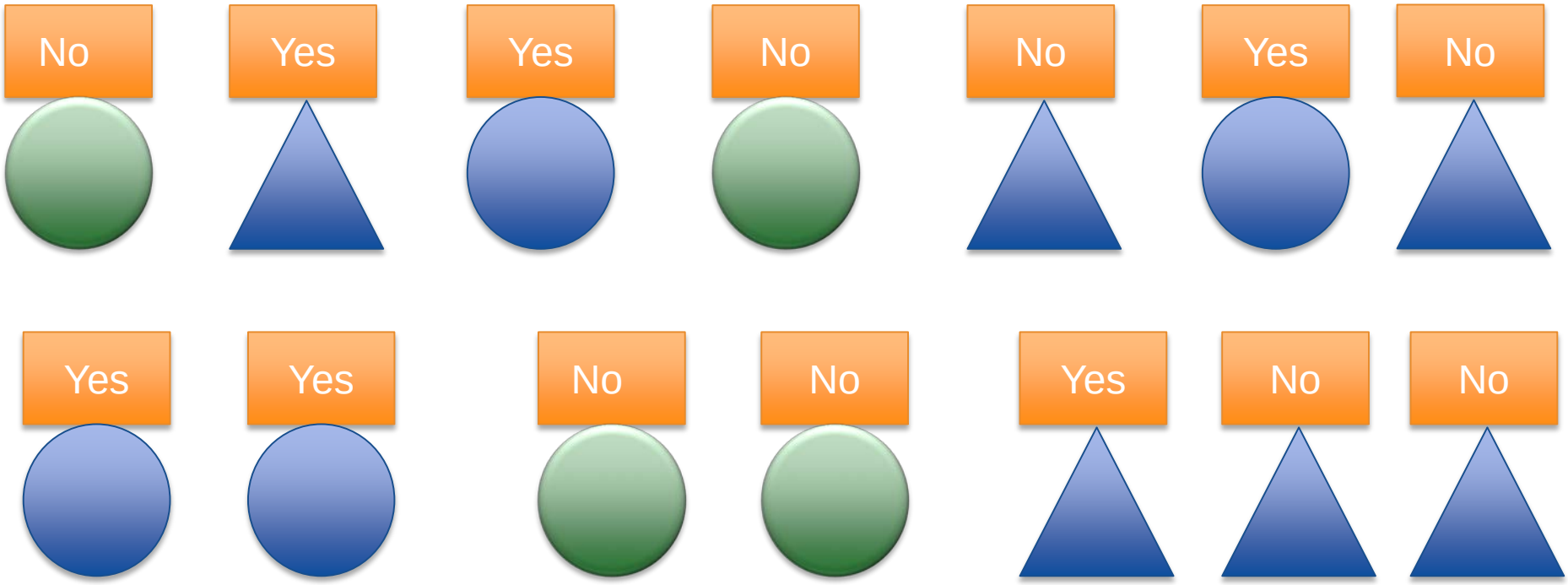


Data Segmentation

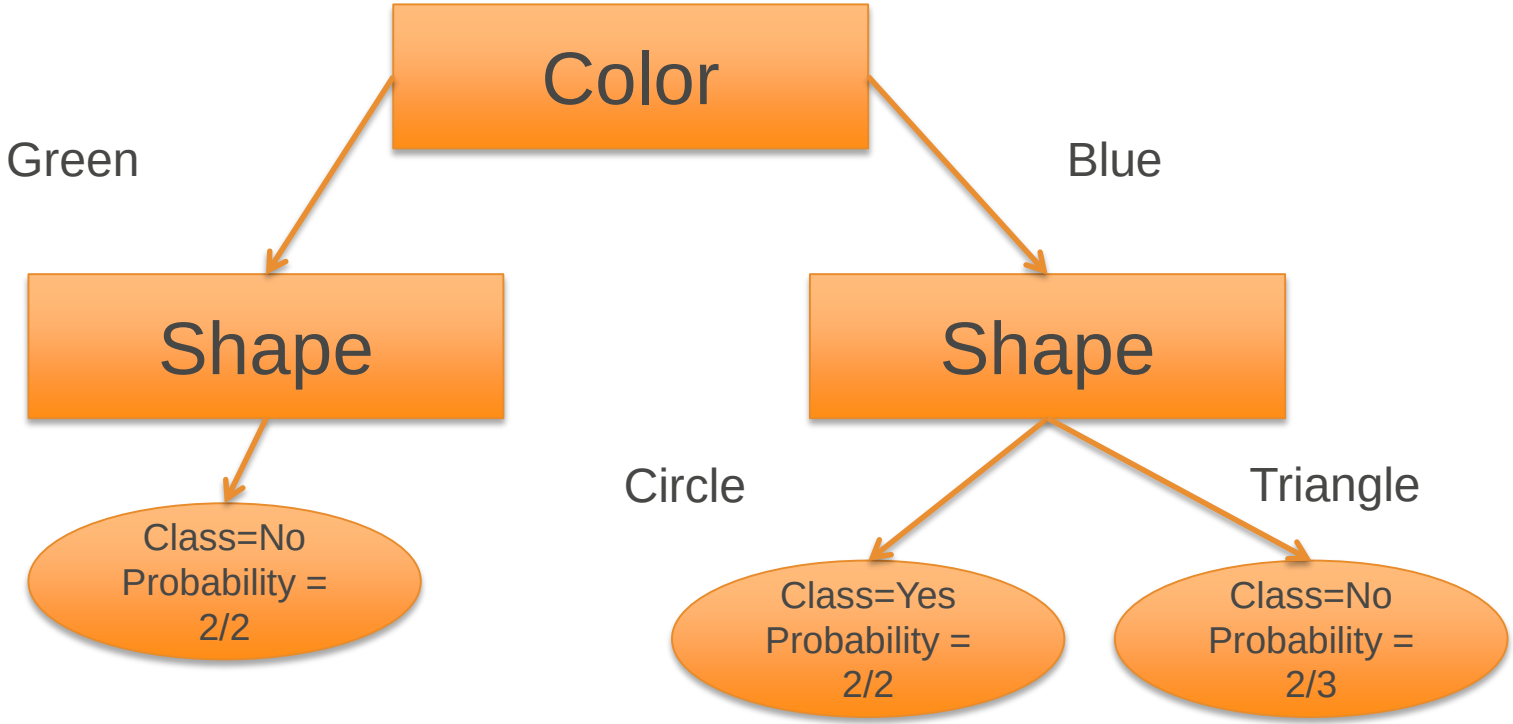
- Divide the population into subgroups that have different values for the target variable
- Look to split on features that create “pure” subgroups



Data Segmentation (Cont'd.)



Tree Induction



Feature Evaluation

- Entropy – Measure of disorder of a dataset

$$entropy = -p_1 \log(p_1) - p_2 \log(p_2) - \cdots$$

- Entropy is useful for determining how well a feature may segment a dataset
- Common to look at entropy when evaluating a dataset for use in training an ML model.

Feature Evaluation (Cont'd.)

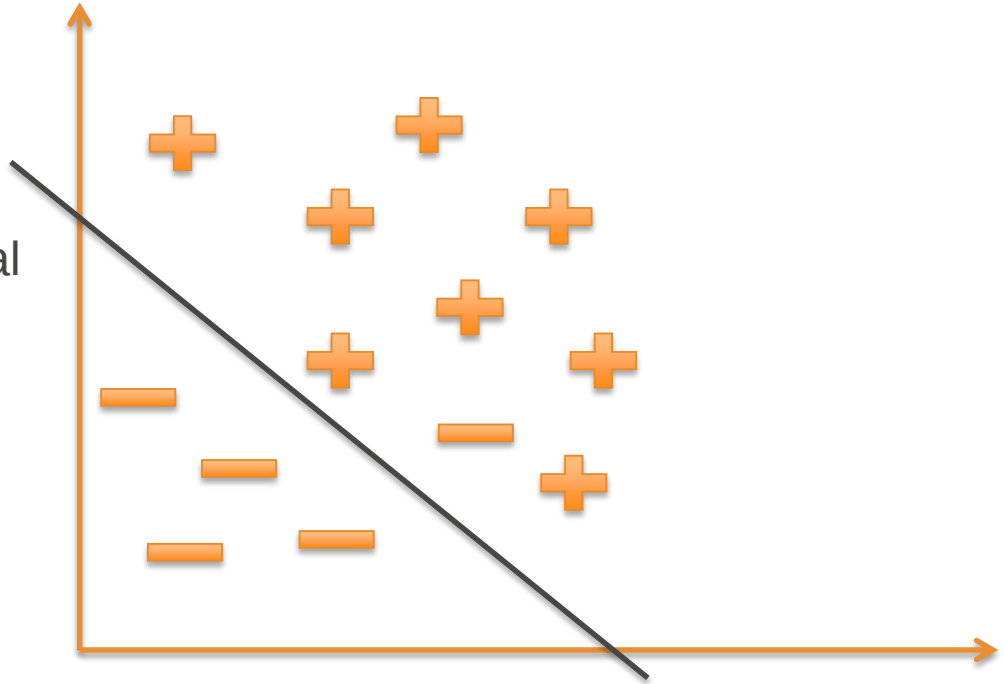
- Information Gain (IG) – Measures how much "purer" a child dataset is after splitting on a feature:

$$IG(parent, children) = entropy(parent) - [p(c_1) \times entropy(c_1) + p(c_2) \times entropy(c_2) + \cdots]$$

- Entropy and Information Gain can be used to determine which attributes are the most "informative" when training an ML model.

Linear Classifier

- “Decision Line” is used to segregate the data
- Mathematical classification, whereas tree induction is a logical classification



Linear Model

$$f(\mathbf{x}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots$$

- Weights are the parameters of the model
- Data mining “fits” the model to a dataset
  - Determine the best set of weights for each feature *x*

Logistic Regression

There can be many ways to segregate a model-what weights are the best?

Minimize errors

- Error is the distance that the evaluated value is from the actual value
- “Least Squares” are often used for error estimation

Logistic Regression (Cont'd.)

Probability Estimation

- *f(x)* values that are farthest from the decision line should have the highest probability of being in one class or another

f(x) is the log-odds that *x* is a member of a class

Log-odds can be used to solve for probability of x belonging to a given class

Overfitting

Models should generalize the data

- A good model should be able to generate an accurate prediction for a previously unseen set of features

Overfitting occurs when the model fits perfectly to the dataset used to train it

- Large numbers of features and complex datasets can cause overfitting

Regularization is used to avoid overfitting

- L1 – reduces the number of features used in the model
- L2 – penalizes very positive or very negative weights, which can fit the function better

Lab : Diving into Machine Learning

Errors or corrections? Email us at aws-course-feedback@amazon.com.
Other questions? Contact us at
https://aws.amazon.com/contact-us/aws-training/.