



Amazon SageMaker Hands-on Guidebook

17th September 2018

김필호 AI Specialist SA / 남궁영환 AI Specialist SA / 서지혜 Partner SA



Table of Contents

Lab 개요	3
목표	3
준비 조건	3
LAB 실습 가이드	4
Module 1: Notebook Instance 생성하기	5
Module 2: 실습용 코드 다운 받기	10
Module 3: 비디오 게임 세일즈 Notebook	12
Module 4: TensorFlow 를 활용한 분산 훈련 Notebook	14
Module 5: 이미지 분류 Notebook	15
Module 6: DeepAR 를 활용한 분산 훈련 Notebook	16
Module 7: Factorization Machine 을 이용한 영화 추천 서비스 Notebook	17
Module 8: Internet-facing 앱 개발	18
Module 8-1: 영어-독어 번역 ML 모델 학습.....	19
Module 8-2: SageMaker Endpoint 호출 Lambda 함수 개발하기.....	24
Module 8-3: AWS API Gateway 와 S3 Static Web Server 를 이용한 웹서비스 연결하기	34
서비스 종료 가이드	45

Lab 개요

Amazon SageMaker 는 데이터 사이언티스트와 개발자들이 쉽고 빠르게 구성, 학습하고 어떤 규모로든 기계 학습된 모델을 배포할 수 있도록 해주는 관리형 서비스 입니다. 이 워크샵을 통해 SageMaker notebook instance 를 생성하고 샘플 Jupyter notebook 을 실습하면서 SageMaker 의 일부 기능을 알아보도록 합니다.

목표

- SageMaker 에 내장된 학습 기능을 사용하여 모델 훈련 Job 을 생성 합니다.
- SageMaker 의 endpoint 기능을 사용하여 생성된 모델이 예측에 사용될 수 있도록 endpoint 를 생성합니다.
- 머신 러닝이 정형 데이터(e.g. CSV 파일)와 비정형 데이터(e.g. 이미지)에 모두 적용 될수 있음을 확인 합니다.

준비 조건

- AWS 계정: AWS IAM, S3, SageMaker 자원을 생성할 수 있는 권한이 필요합니다.
- AWS Region: SageMaker 는 지원되는 region 은 <https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/> 에서 확인하실 수 있습니다. 이번 실습은 **Seoul (ap-northeast-2) region** 에서 실행 합니다.
- Browser: 최신 버전의 **Chrome, Firefox** 를 사용하세요.

※ 주의 사항: Notebook 안의 Cell 에서 코드 실행후 결과 값이 나오는 데는 수 초가 걸립니다. 훈련 Job 을 실행하는 경우 수 분이 걸릴 수도 있습니다. 실습 완료 후에는 아래 가이드에 따라 생성된 자원을 꼭 종료/삭제해 주세요.

LAB 실습 가이드

실습은 총 4 개 모듈로 구성되어 있습니다. 1 번 완료후 2 번을 순서대로 진행하셔야 합니다. 3 번, 4 번, 5 번, 6 번 모듈은 원하는 순서대로 진행하실 수 있습니다.

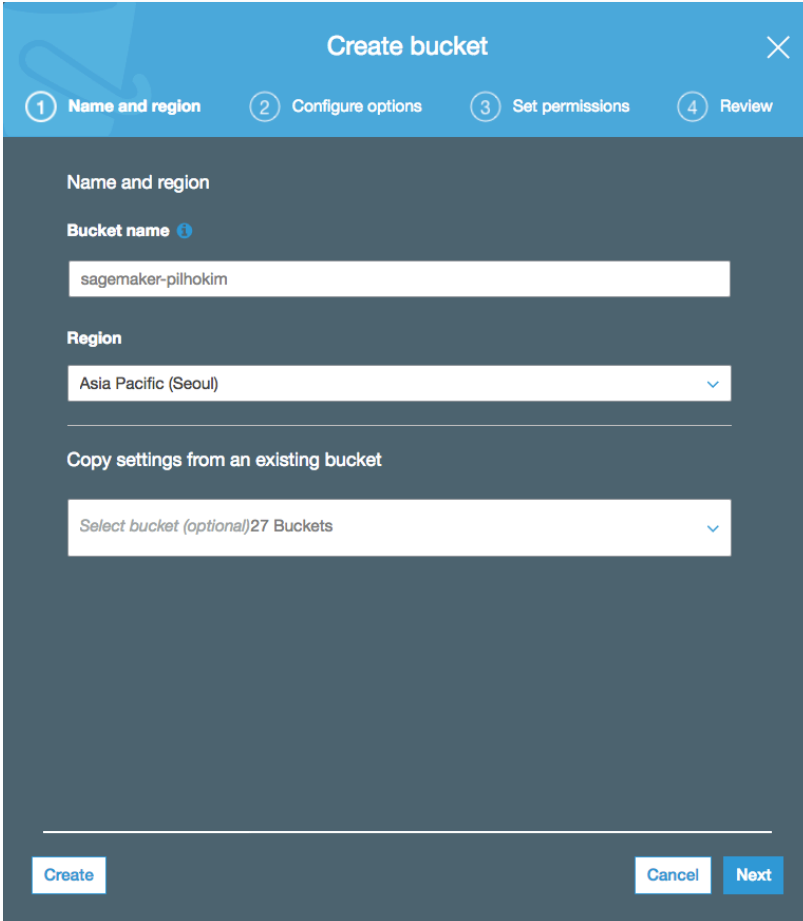
1. Notebook Instance 생성하기
2. 실습용 코드 다운받기
3. 비디오 게임 세일즈 Notebook
4. TensorFlow 를 활용한 분산 훈련 Notebook
5. 이미지 분류 Notebook
6. Internet-facing 앱 개발
7. DeepAR-HomeElectric 적용 예

Module 1: Notebook Instance 생성하기

1. S3 Bucket 생성하기

SageMaker 는 S3 를 데이터와 모델 저장소로 사용합니다. 여기서는 해당 목적으로 S3 Bucket 을 생성합니다.

- 1) AWS 관리 콘솔 (<https://console.aws.amazon.com/>)에 Sign in 합니다.
- 2) AWS Services 리스트에서 S3 로 이동합니다.
- 3) "+ Create Bucket" 버튼을 선택합니다.
- 4) 아래 내용 설정 후 화면 왼쪽 아래 **Create** 클릭합니다.
 - Bucket name: sagemaker-{userid} [반드시 고유한 값 설정]
 - Region : Asia Pacific (Seoul)



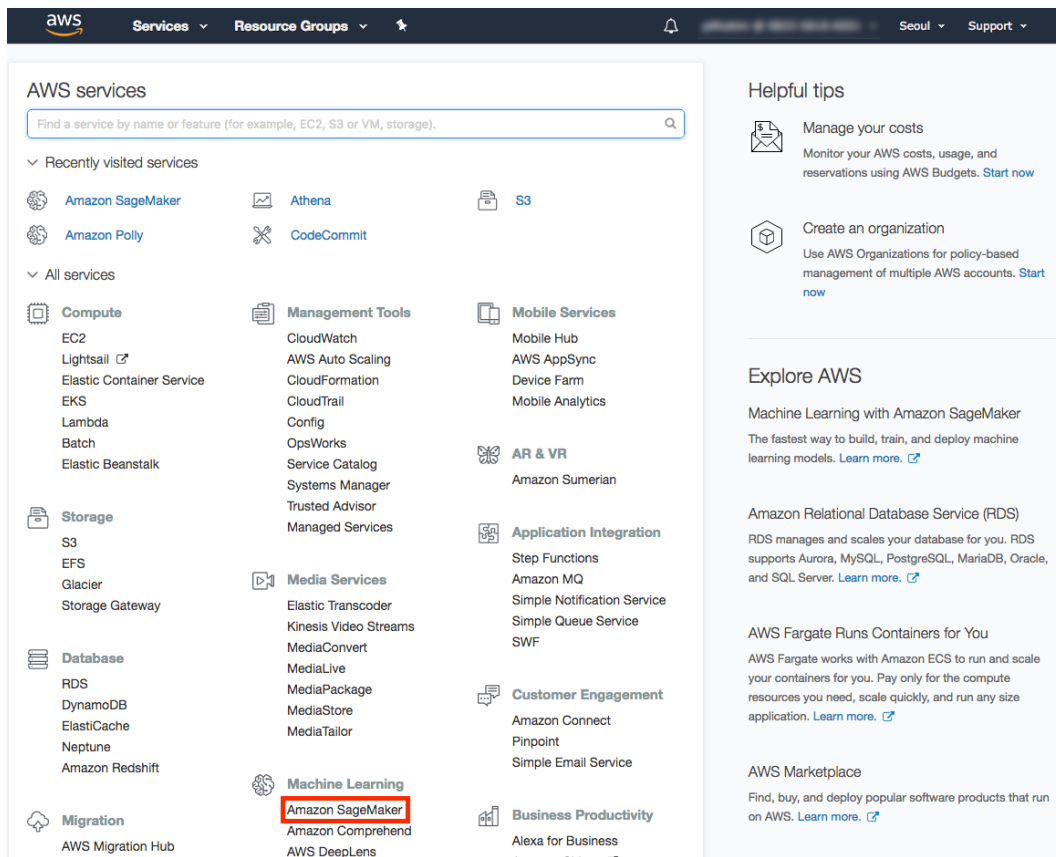
The screenshot shows the 'Create bucket' wizard in the AWS console. The title bar is blue with a close button (X) on the right. Below the title bar is a progress indicator with four steps: 1. Name and region (active), 2. Configure options, 3. Set permissions, and 4. Review. The main content area is dark blue and contains the following fields:

- Name and region**
 - Bucket name**: A text input field containing 'sagemaker-pilhokim'.
 - Region**: A dropdown menu showing 'Asia Pacific (Seoul)'.
- Copy settings from an existing bucket**: A dropdown menu showing 'Select bucket (optional) 27 Buckets'.

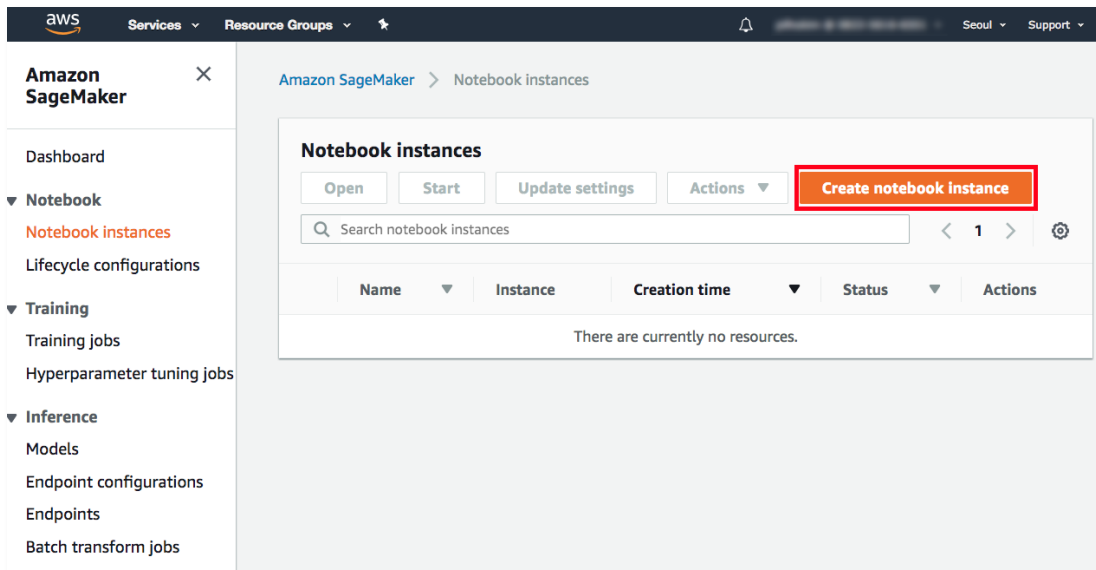
At the bottom of the form, there are three buttons: 'Create' (highlighted in blue), 'Cancel', and 'Next'.

2. Notebook instance 생성

1) AWS 관리 콘솔에서 오른쪽 상단에서 Seoul Region 선택 후 AWS Services 리스트에서 Amazon SageMaker 서비스를 선택합니다.



2) 새로운 Notebook instance 를 생성하기 위해 왼쪽 패널 메뉴 중 **Notebook Instances** 선택 후 오른쪽 상단의 **Create notebook instance** 버튼을 클릭 합니다.



3) Notebook instance 이름으로 **[First Name]-[Last Name]-workshop** 으로 넣은 뒤 **ml.m4.xlarge** 인스턴스 타입을 선택 합니다.

Amazon SageMaker > Notebook instances > Create notebook instance

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. [Learn more](#)

Notebook instance settings

Notebook instance name

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the [AmazonSageMakerFullAccess](#) IAM policy attached.

VPC - optional

Your notebook instance will be provided with SageMaker provided internet access because a VPC setting is not specified.

Lifecycle configuration - optional

Customize your notebook environment with default scripts and plugins.

Encryption key - optional

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

▼ Tags - optional

4) IAM role 은 **Create a new role** 을 선택하고, 생성된 팝업 창에서는 **S3 buckets you specify – optional** 밑에 **Specific S3 Bucket** 을 선택 합니다. 그리고 텍스트 필드에 위에서 만든 S3 bucket 이름(예: sagemaker-xxxxx)을 선택 합니다. 이후 **Create role** 을 클릭합니다.

true"', and 'S3 bucket with a Bucket Policy allowing access to SageMaker'. At the bottom, there are 'Cancel' and 'Create role' buttons."/>

Create an IAM role

Passing an IAM role gives Amazon SageMaker permission to perform actions in other AWS services on your behalf. Creating a role here will grant permissions described by the [AmazonSageMakerFullAccess](#) IAM policy to the role you create.

The IAM role you create will provide access to:

- S3 buckets you specify - *optional*
 - Specific S3 buckets

Example: bucket-name-1, bucket-name-2, bi

Comma delimited. ARNs, "*" and "/" are not supported.
 - Any S3 bucket

Allow users that have access to your notebook instance access to any bucket and its contents in your account.
 - None
- Any S3 bucket with "sagemaker" in the name
- Any S3 object with "sagemaker" in the name
- Any S3 object with the tag "sagemaker" and value "true" [See Object tagging](#)
- S3 bucket with a Bucket Policy allowing access to SageMaker [See S3 bucket policies](#)

5) 다시 Create Notebook instance 페이지로 돌아온 뒤 **Create notebook instance** 를 클릭합니다.

3. Notebook Instance 접근하기

1) 서버 상태가 **InService** 로 바뀔 때까지 기다립니다. 보통 5 분정도의 시간이 소요 됩니다.

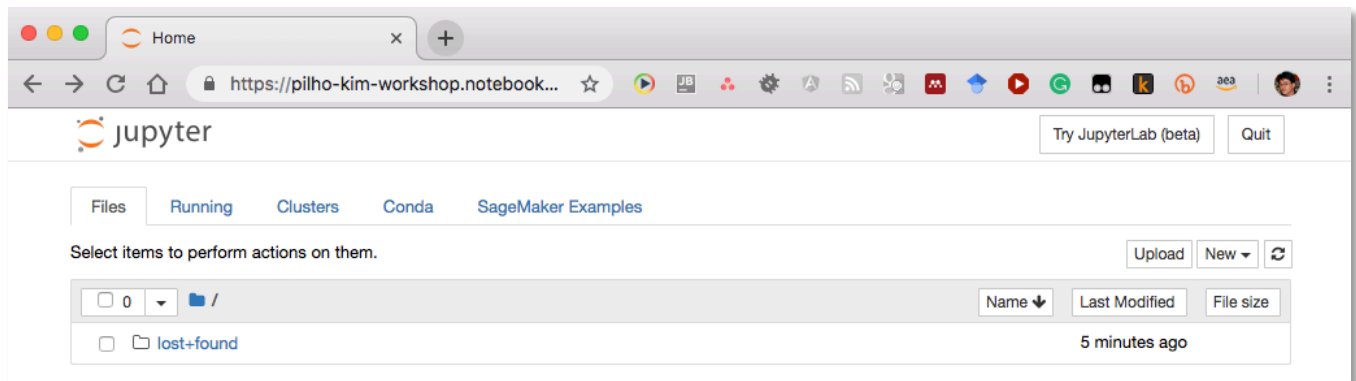
Amazon SageMaker > Notebook instances

Notebook instances

Search notebook instances < 1 > ⚙

	Name	Instance	Creation time	Status	Actions
<input type="radio"/>	pilho-kim-workshop	ml.m4.xlarge	Sep 17, 2018 08:24 UTC	<input checked="" type="radio"/> InService	Open Stop

2) Open 을 클릭하면 방금 생성한 notebook instance 의 Jupyter 홈페이지로 이동하게 됩니다.



Module 2: 실습용 코드 다운 받기

SageMaker 의 Jupyter 노트북도 Linux 기반의 서버입니다. Jupyter 노트북에서 서버의 Terminal 을 바로 실행하는 기능을 제공하고 있습니다. Figure 1 와 같이 Terminal 을 선택합니다.

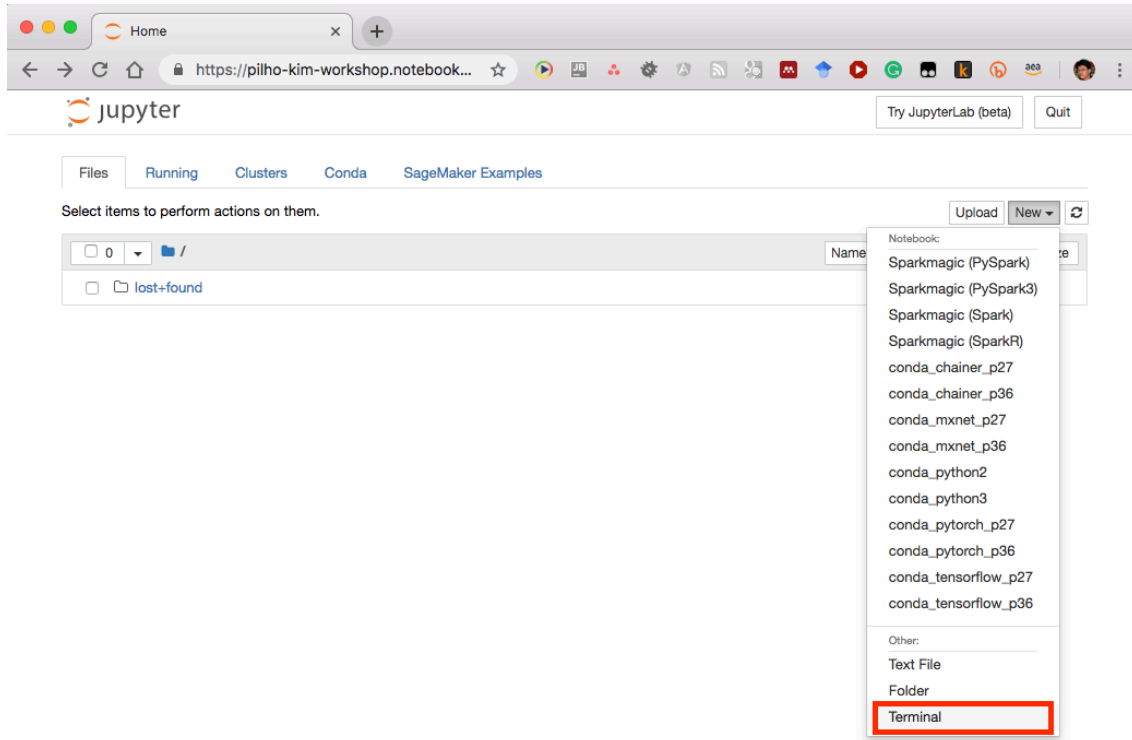


Figure 1. SageMaker 노트북 서버에 접속하기 위한 Terminal 실행 화면.

Figure 2 과 같이 터미널이 실행되면 아래의 명령어들을 입력해서 실행합니다.

```
cd SageMaker/
git clone https://github.com/pilhokim/ai-ml-workshop
```

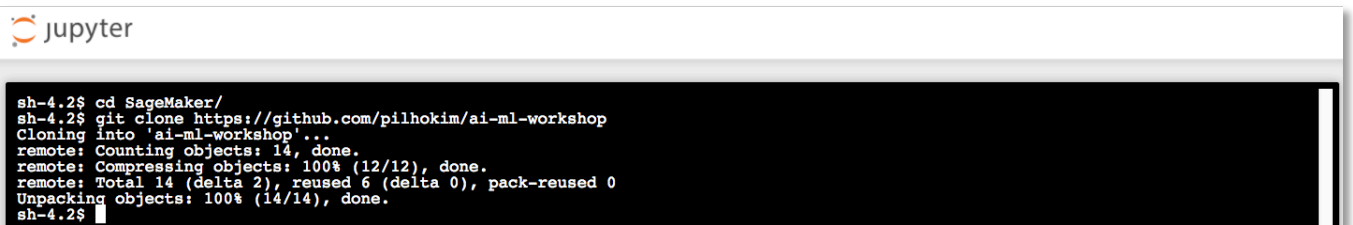


Figure 2. git 사이트에서 실습 코드 다운 받기.

코드를 다운 받고 난 후 Jupyter 노트북을 갱신 하면 (오른쪽 상단의 Refresh 아이콘을 클릭하세요) 새롭게 다운 받은 코드 폴더가 보입니다 (Figure 3 참조).

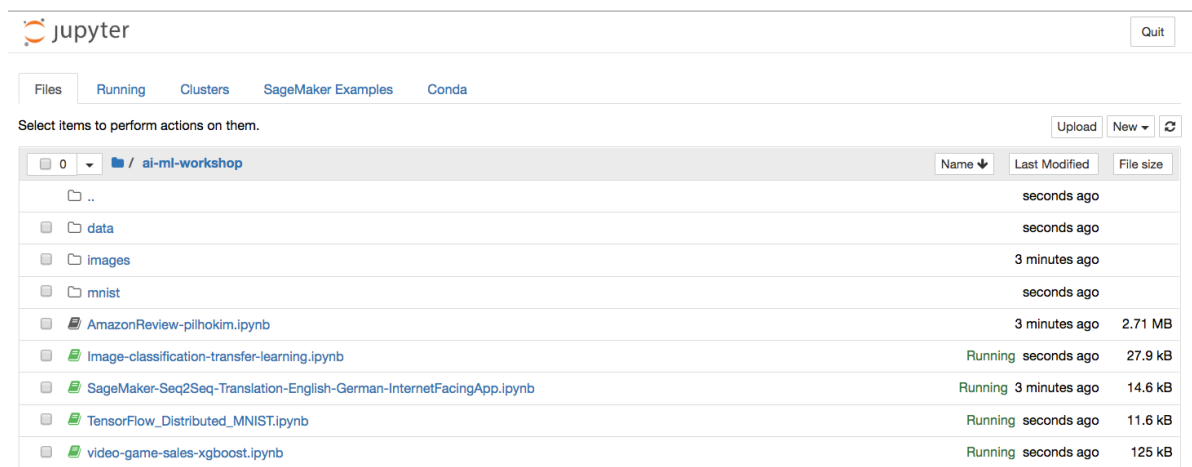


Figure 3. 새로 다운 받은 폴더 생성 확인 화면.

Module 3: 비디오 게임 세일즈 Notebook

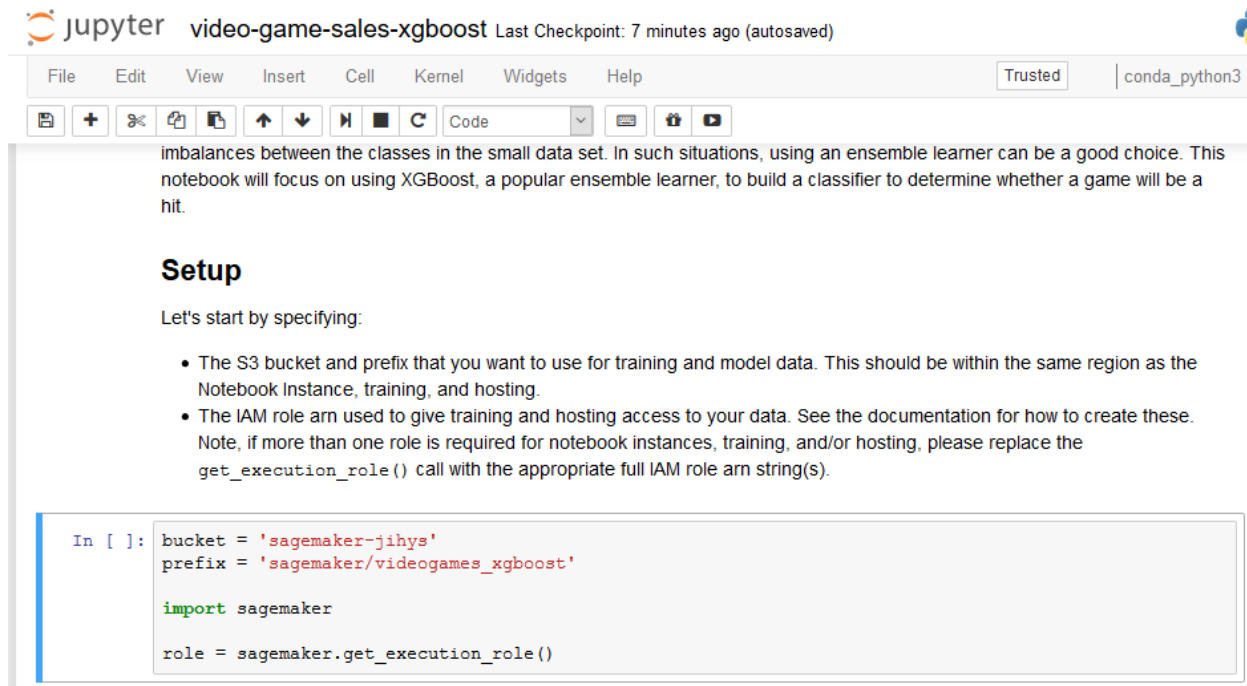
이 모듈에서는 Jupyter notebook 예제를 통해 어떻게 아마존이 제공하는 알고리즘을 SageMaker 에서 사용할 수 있는지 알아 봅니다. 특히 SageMaker 버전의 XGBoost 알고리즘을 사용하게 되는데, XGBoost 는 Gradient boosted decision tree 알고리즘을 구현한 유명하고 효율적인 오픈 소스버전입니다. Gradient boosting 은 supervised learning 알고리즘 중에 하나로 단순하고 weak 한 모델들의 예측치를 결합하여 타겟 변수를 예측합니다. XGBoost 는 다양한 종류의 데이터 타입과, 관계, 분산을 처리할 수 있기 때문에 많은 머신 러닝 경진 대회에서 우수한 결과를 낸 알고리즘 입니다. 이 알고리즘은 관계형 데이터 베이스 또는 Flat 파일등과 같은 정형 데이터를 다룰 경우 바로 사용 할 수 있는 알고리즘입니다.

실습을 위해서 현재 설치되어 있는 SageMaker 의 Jupyter 노트북의 예제들 중 아래의 디렉토리에 위한 Jupyter 노트북을 실행하시면 됩니다.

/ai-ml-workshop/2018-09/module3-video-game-sales-xgboost.ipynb

1. 첫번째 Cell 에서 **bucket= '<your_s3_bucket_name_here>'** 라인에서

<your_s3_bucket_name_here>을 모듈 1 에서 만든 S3 bucket 이름(예: sagemaker-xxxxx)을 적습니다. S3://...와 같은 경로 이름은 적지 않습니다.



imbalances between the classes in the small data set. In such situations, using an ensemble learner can be a good choice. This notebook will focus on using XGBoost, a popular ensemble learner, to build a classifier to determine whether a game will be a hit.

Setup

Let's start by specifying:

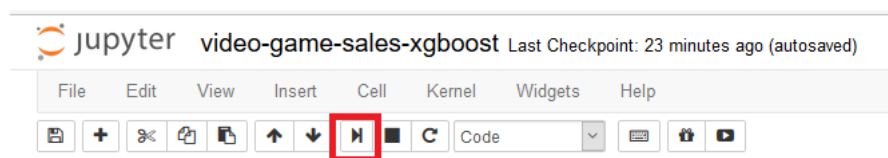
- The S3 bucket and prefix that you want to use for training and model data. This should be within the same region as the Notebook Instance, training, and hosting.
- The IAM role arn used to give training and hosting access to your data. See the documentation for how to create these. Note, if more than one role is required for notebook instances, training, and/or hosting, please replace the `get_execution_role()` call with the appropriate full IAM role arn string(s).

```
In [ ]: bucket = 'sagemaker-jihys'
        prefix = 'sagemaker/videogames_xgboost'

import sagemaker

role = sagemaker.get_execution_role()
```

2. Jupyter notebook 은 코드와 주석을 같이 저장합니다. Jupyter notebook 에는 두 가지의 Cell(Code Cell 과 markdown Cell)이 있습니다. Code 를 실행하려면 실행 버튼을 클릭합니다. (Control+Enter 도 동일한 기능이며, 실제 사용하는 실행 후 셀을 이동하는 Shift+Enter 가 더 편리합니다.)



3. Code 가 실행되면 Code Cell 왼쪽의 "In []" 라는 부분이 "In [*]"로 변경이 되고 완료시에는 실행 순서를 나타내는 숫자로 변경 됩니다.

※ 모델 훈련에는 약 8 분 정도가 소요됩니다.

※ 코드는 Code Cell 에 나타난 순서대로 실행하고 반복 작업을 피하기 위해서 한 번만 실행합니다. 같은 훈련 job cell 을 반복 실행하게 되면 두 개의 훈련 job 을 실행하게 되어 서비스 제한을 넘을 수도 있습니다.

Module 4: TensorFlow 를 활용한 분산 훈련 Notebook

이 모듈에서는 [MNIST Database](#) 에서 손으로 쓴 숫자의 이미지 데이터를 활용하여 SageMaker 에서 어떻게 분산 훈련을 실행하는지 설명합니다. [TensorFlow MNIST Example](#) 에 기반한 Convolutional Neural Network model 을 활용합니다.

이 모델을 통해 데이터 전처리 작업, 모델 훈련, SageMaker hosted endpoint 생성, 훈련된 모델을 endpoint 에 실제로 적용 하기 위해 어떻게 Jupyter notebook 과 SageMaker Python SDK 를 사용하는지 설명합니다.

생성된 모델은 실제로 사용자가 그려 넣은 숫자가 무엇인지 예측합니다. 이 예제에서는 TensorFlow 를 사용해 자신의 코드를 가져와 실행하는 것 뿐만 아니라, SageMaker 에서 모델 훈련을 위해 여러 대의 인스턴스 클러스터를 얼마나 쉽게 생성할 수 있는지 보여 줍니다.

실습을 위해서 현재 설치되어 있는 SageMaker 의 Jupyter 노트북의 예제들 중 아래의 디렉토리에 위한 Jupyter 노트북을 실행하시면 됩니다.

`/ai-ml-workshop/2018-09/module4-TensorFlow_Distributed_MNIST.ipynb`

※ 모델 훈련에는 약 8 분 정도가 소요됩니다.

Module 5: 이미지 분류 Notebook

이 모듈에서는 이미지 분류 예제를 실행합니다. 특히, 아마존에서 제공하는 이미지 분류 알고리즘을 활용합니다. 이 알고리즘은 Supervised learning 알고리즘으로 이미지를 인풋으로 받아 여러 개의 아웃풋 카테고리 중에서 하나로 분류 합니다.

이 알고리즘은 Convolutional Neural Network 중의 하나인 ResNet 을 활용하는데, 처음 부터 이를 이용한 훈련을 할수도 있고, 충분한 수의 훈련 이미지가 없을 때 transfer learning 을 써서 훈련 할 수도 있습니다.

이 실습에서는 직접 신경망을 설계하거나 구현하지 않고, 신경망이나 이미지 분류에 대한 지식이 없더라도 SageMaker 의 이미지 분류 알고리즘이 얼마나 쉽게 활용 될 수 있는지 알아봅니다.

실습을 위해서 현재 설치되어 있는 SageMaker 의 Jupyter 노트북의 예제들 중 아래의 디렉토리에 위한 Jupyter 노트북을 실행하시면 됩니다.

`/ai-ml-workshop/2018-09/module5-Image-classification-transfer-learning.ipynb`

※ 이 모델을 훈련하는데는 약 10 분이 소요됩니다. Transfer learning 이 사용되기 때문에 비교적 짧은시간이 걸립니다.

※ 이 모델을 훈련하는데는 ml.p2.8xlarge 이 사용됩니다. 만약 본인의 account 설정에 service limit 이 걸려있다면 http://docs.amazonaws.cn/en_us/general/latest/gr/aws_service_limits.html 의 안내에 따라 "Amazon SageMaker Training"의 limit increase 를 실습 며칠전에 미리 신청하셔야 합니다.

To request a limit increase

1. Open the [AWS Support Center](#) page, sign in if necessary, and choose **Create case**.
2. For **Regarding**, choose **Service Limit Increase**.
3. Complete the form. If this request is urgent, choose **Phone** as the method of contact instead of **Web**.
4. Choose **Submit**.

Module 6: DeepAR 를 활용한 분산 훈련 Notebook

이 모듈에서는 DeepAR 에 대한 소개와 가정 전력소모 데이터에 대한 예측 모델을 만드는 과정입니다. 본 실습을 통해:

- Python 을 활용한 데이터 정제
- DeepAR 모델 훈련 및 배포
- DeepAR 의 Advanced features 에 대한 활용

을 다루게 됩니다.

실습을 위해서 현재 설치되어 있는 SageMaker 의 Jupyter 노트북의 예제들 중 아래의 디렉토리에 위한 Jupyter 노트북을 실행하시면 됩니다.

`/ai-ml-workshop/2018-09/module6-DeepAR-HomeElectric.ipynb`

Module 7: Factorization Machine 을 이용한 영화 추천 서비스 Notebook

이 모듈에서는 Factorization Machines 에 대한 소개와 이를 이용한 영화 추천 서비스를 만드는 과정입니다. 본 실습을 통해:

- Factorization Machines 알고리즘을 위한 데이터 준비 과정
- Factorization Machines 모델 훈련 및 배포

을 다루게 됩니다.

실습을 위해서 현재 설치되어 있는 SageMaker 의 Jupyter 노트북의 예제들 중 아래의 디렉토리에 위한 Jupyter 노트북을 실행하시면 됩니다.

/ai-ml-workshop/2018-09/module8-Movie recommendation on Amazon SageMaker Using Factorization Machines-5thSep2018.ipynb

Module 8: Internet-facing 앱 개발

Amazon SageMaker 는 데이터 사이언티스트와 개발자들이 쉽고 빠르게 구성, 학습하고 어떤 규모 로든 기계 학습된 모델을 배포할 수 있도록 해주는 관리형 서비스 입니다. 이 워크샵을 통해 Sagemaker notebook instance 를 생성하고 샘플 Jupyter notebook 을 실행하면서 SageMaker 의 일부 기능을 알아보도록 합니다.

이 모듈에서는 영어를 독일어로 변환하는 SageMaker 의 Sequence-to-Sequence 알고리즘을 이용한 언어번역기를 학습해보고 이 서비스를 인터넷을 통해 활용할 수 있는 방법에 대해 실습해 보겠습니다.

본 Hands-on 에서는 SageMaker 에서 생성한 Endpoint inference service 를 웹 상에서 호출하기 위해 AWS Lambda 와 AWS API Gateway 를 Figure 4 과 같은 데모를 만들어 보겠습니다.

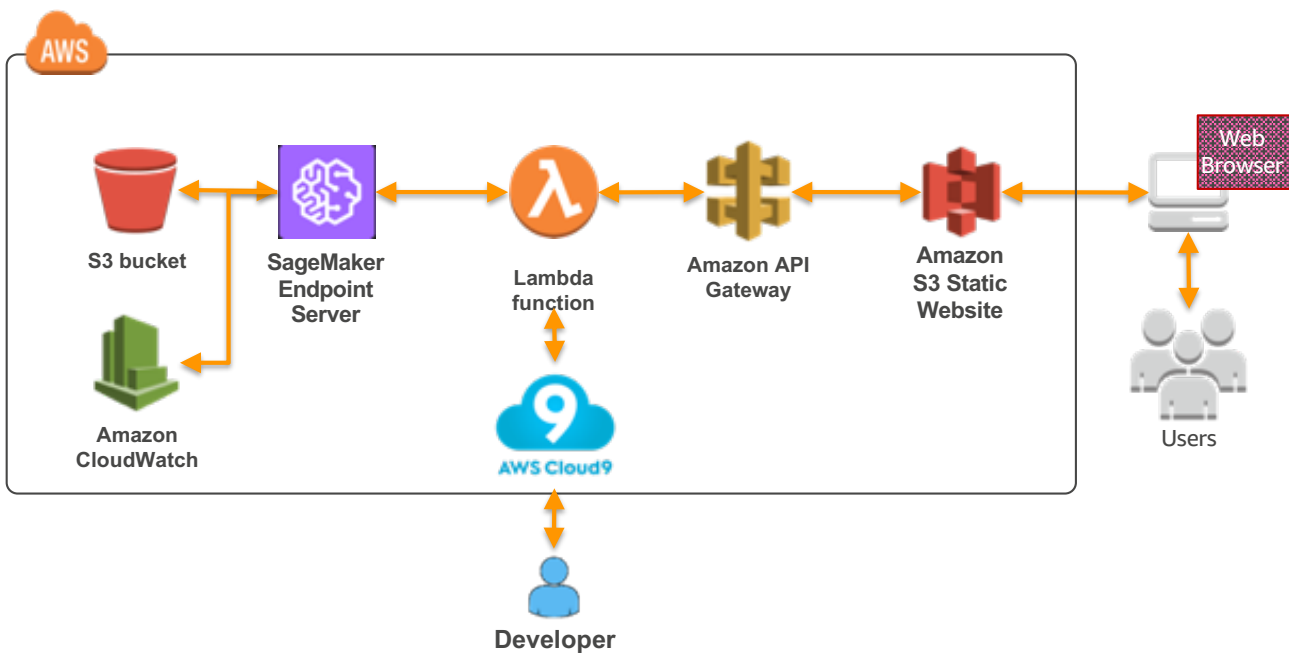


Figure 4. SageMaker Internet-facing App Data Flow.

Figure 4 에서는 SageMaker 의 기능 데모를 위해 가장 간략한 구조를 채택하고 있습니다. 예를 들어 [Amazon S3 의 Static Website 에 다른 도메인 이름을 지정하기 위한 Route 53 서비스](#)나 [캐싱 서비스를 위한 CloudFront](#) 등의 서비스는 실제 비즈니스 적용 시에는 고려 되어야할 서비스입니다.

전제 Lab 시간은 일반 사용자의 경우 한시간에서 한시간 30 분정도 소요 예상 됩니다.



Module 8-1: 영어-독어 번역 ML 모델 학습

Sequence-to-Sequence 알고리즘 노트북 열기

SageMaker 가 지원하는 Seq2Seq 알고리즘은 MXNet 기반으로 개발된 [Sockeye](#) 알고리즘을 기반으로 개발된 최신의 Encoder-decoder 구조를 구현한 것으로 문서자동요약이나 언어 번역 서비스에 적용할 수 있습니다.

실습을 위해서 현재 설치되어 있는 SageMaker 의 Jupyter 노트북의 예제들 중 아래의 디렉토리에 위한 Jupyter 노트북을 실행하시면 됩니다 (Figure 5 참조).

/ai-ml-workshop/2018-09/module7-SageMaker-Seq2Seq-Translation-English-German-InternetFacingApp.ipynb

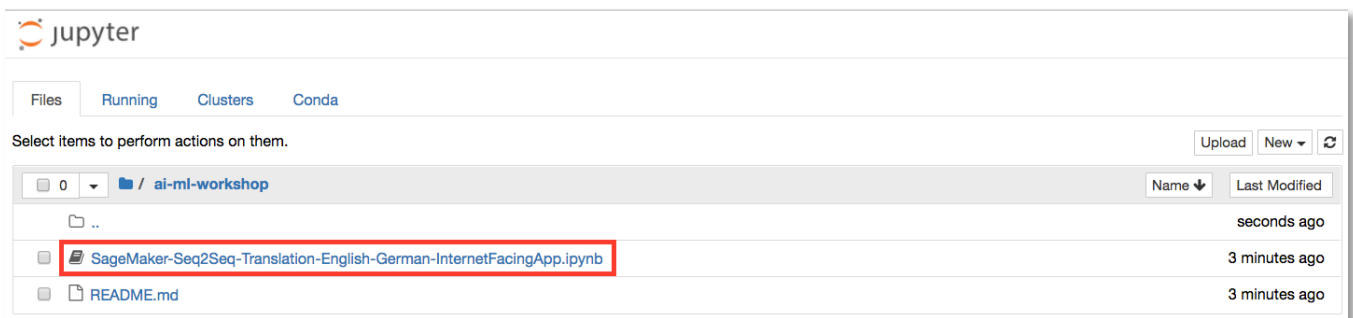


Figure 5. Seq2Seq 노트북 디렉토리 위치.



Figure 6. 노트북 화면.

노트북에 대한 설명

본 노트북은 아래에 위치한 예제 노트북의 수정된 버전으로 미리 학습된 머신 러닝 모델을 사용하도록 바뀌었습니다.

/sample-notebooks/introduction_to_amazon_algorithms/seq2seq_translation_en-de/SageMaker-Seq2Seq-Translation-English-German.ipynb

상기 노트북은 빠른 학습 시간을 위해 Figure 7 와 같이 전체 데이터 중 첫번째 10000 개의 데이터의 대해서만 학습을 해서 Seq2Seq 알고리즘의 사용방법을 소개하고 있습니다.

Since training on the whole dataset might take several hours/days, for this demo, let us train on the **first 10,000 lines only**. Don't run the next cell if you want to train on the complete dataset.

```
In [5]: !head -n 10000 corpus.tc.en > corpus.tc.en.small
!head -n 10000 corpus.tc.de > corpus.tc.de.small
```

Figure 7. 샘플 데이터 선택 화면.

Figure 8 는 다운받은 corpus 의 실제 데이터 내용으로 영어 및 독일어 데이터가 어떻게 문장 대 문장으로 매핑 되고 있는지를 보여주고 있습니다.



영문 데이터 (corpuc.tc.en.small 내용)

독일어 데이터(corpuc.tc.de.small 내용)

Figure 8. 번역기 학습을 위한 영문 자료와 독일어 자료 비교 화면.

실제로는 10000 개의 샘플 문장으로 훈련한 번역기는 좋은 결과를 보여줄 수 없습니다. 그렇지만 전체 데이터 학습을 위해서는 선택하시는 SageMaker 의 서버 Instance Type 에 따라 다르지만 수시간에서 수일의 장시간이 소요될 수 있습니다. 따라서 이 노트북의 개발자들은 좀더 나은



품질의 번역 결과 체험을 원하시는 사용자들 위해 전체 데이터에 이미 훈련이 된 모델을 공유하고 있습니다.

이 Pre-trained model 을 사용하기 위해서는 노트북의 코드 중 Endpoint Configuration 직전의 코드를 아래와 같이 수정해서 이미 훈련된 모델을 다운로드 한 다음 본인의 S3 버킷으로 업로드 하시면 됩니다. 이때 Jupyter 노트북 마지막 줄의 `sage.delete_endpoint` 는 데모를 계속 진행하기 위해 실행하지 않습니다. 이를 위해 이번에는 가장 마지막 줄에 있는 코드를 주석 처리하겠습니다.

Stop / Close the Endpoint (Optional)

Finally, we should delete the endpoint before we close the notebook.

```
In [ ]: # sage.delete_endpoint(EndpointName=endpoint_name)
```

Figure 9. `delete_endpoint` 함수 콜 코멘트 처리 화면.

Pre-trained 모델을 사용 하기 위한 노트북 수정

노트북에서 하단의 S3 bucket 이름에 상기 생성한 S3 이름을 입력하시고 우측의 예와 비슷한 형식으로 prefix 를 입력하시면 됩니다 (Figure 11 참조).

```
# S3 bucket and prefix
bucket = '<your_s3_bucket_name_here>'
prefix = 'sagemaker/<your_s3_prefix_here>' # E.g. 'sagemaker/seq2seq/eng-german'
```

Figure 10. 노트북 S3 버킷 이름 및 prefix 수정 전 화면.

```
In [1]: # S3 bucket and prefix
bucket = 'sagemaker-pilho-hands-on'
prefix = 'sagemaker/seq2seq/eng-german' # E.g. 'sagemaker/seq2seq/eng-german'
```

Figure 11. S3 버킷 및 prefix 수정 후 화면 예제. 본인의 S3 버킷 이름으로 수정하셔야 합니다.

노트북 실행 방법

이제 노트북 전체를 실행할 준비가 되었습니다. Jupyter 노트북을 실행하는 방법은 코드가 있는 셀을 클릭으로 선택하신 후 Shift-enter 키를 누르시거나 또는 Jupyter 노트북 상단의 툴바에서 "Run cell, select below" 버튼을 클릭하셔도 됩니다.

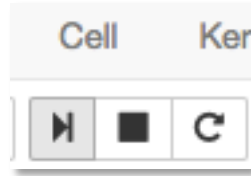


Figure 12. Jupyter 노트북 셀 실행 톨바.

전체 실행 과정은 약 12 분에서 15 분 정도 소요 됩니다. 각각의 셀을 실행시키면서 셀 하단에 표시되는 처리결과들을 확인해 보시기 바랍니다.

노트북 코드 중 "Create endpoint configuration" 셀에서 현재 *InstanceType* 이 'ml.m4.large' 로 되어 있습니다 (Figure 13 참조). Seq2Seq 알고리즘은 Neural network 기반이기 때문에 ml.p2.xlarge (GPU) instance 를 이용하실 수 있지만 본 실습에서는 Free tier 가 지원되는 ml.m4.xlarge 를 사용하고 있습니다. ml.t2.* instance 는 time-out 문제가 발생할 수 있으므로 본 실습에서는 사용하지 않습니다.

```
In [12]: from time import gmtime, strftime

endpoint_config_name = 'Seq2SeqEndpointConfig-' + strftime("%Y-%m-%d-%H-%M-%S", gmtime())
print(endpoint_config_name)
create_endpoint_config_response = sage.create_endpoint_config(
    EndpointConfigName = endpoint_config_name,
    ProductionVariants=[{
        'InstanceType': 'ml.m4.xlarge',
        'InitialInstanceCount': 1,
        'ModelName': model_name,
        'VariantName': 'AllTraffic'}])

print("Endpoint Config Arn: " + create_endpoint_config_response['EndpointConfigArn'])

Seq2SeqEndpointConfig-2018-03-24-08-35-50
Endpoint Config Arn: arn:aws:sagemaker:us-east-1:082256166551:endpoint-config/seq2seqendpoint
config-2018-03-24-08-35-50
```

Figure 13. Endpoint configuration 화면.

노트북 코드 중 "Create endpoint" 셀은 새로운 서버를 설치하고 실행 코드를 설치하는 과정이므로 본 노트북에서는 가장 많은 시간 (약 10~11 여분)이 소요 되는데 아래와 같은 메시지를 확인하시면 다음 모듈로 진행하시면 됩니다 (Figure 14 참조).

Endpoint creation ended with EndpointStatus = InService



Create endpoint

Lastly, we create the endpoint that serves up model, through specifying the name and configuration defined above. The end result is an endpoint that can be validated and incorporated into production applications. This takes 10-15 minutes to complete.

```
In [21]: %%time
import time

endpoint_name = 'DEMO-Seq2SeqEndpoint-' + strftime("%Y-%m-%d-%H-%M-%S", gmtime())
print(endpoint_name)
create_endpoint_response = sage.create_endpoint(
    EndpointName=endpoint_name,
    EndpointConfigName=endpoint_config_name)
print(create_endpoint_response['EndpointArn'])

resp = sage.describe_endpoint(EndpointName=endpoint_name)
status = resp['EndpointStatus']
print("Status: " + status)

# wait until the status has changed
sage.get_waiter('endpoint_in_service').wait(EndpointName=endpoint_name)

# print the status of the endpoint
endpoint_response = sage.describe_endpoint(EndpointName=endpoint_name)
status = endpoint_response['EndpointStatus']
print('Endpoint creation ended with EndpointStatus = {}'.format(status))

if status != 'InService':
    raise Exception('Endpoint creation failed.')

DEMO-Seq2SeqEndpoint-2018-03-13-06-35-48
arn:aws:sagemaker:us-east-1:082256166551:endpoint/demo-seq2seqendpoint-2018-03-13-06-35-48
Status: Creating
Endpoint creation ended with EndpointStatus = InService
CPU times: user 92 ms, sys: 0 ns, total: 92 ms
Wall time: 10min 32s
```

Figure 14. SageMaker Endpoint 생성 결과 화면.

노트북 가장 하단의 `delete_endpoint`는 주석 처리 되어 있어야 endpoint 서버가 다음 실습을 위해 계속 운용될 수 있습니다. 만약에 실행 전에 수정하셨다면 “**Create endpoint**” 부분의 코드를 다시 실행하시기 바랍니다.

Module 8-2: SageMaker Endpoint 호출 Lambda 함수 개발하기

본 모듈에서는 방금 생성한 SageMaker 의 Inference service 를 호출하는 Lambda 함수를 개발해 보겠습니다.

Lambda 함수 생성하기

1. AWS 콘솔에서 Lambda 를 선택 (<https://console.aws.amazon.com/lambda>)
2. "Create function" 선택 (Figure 15 참조)

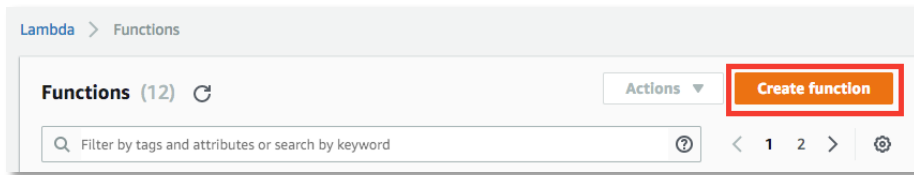


Figure 15. Lambda 함수 생성 화면.

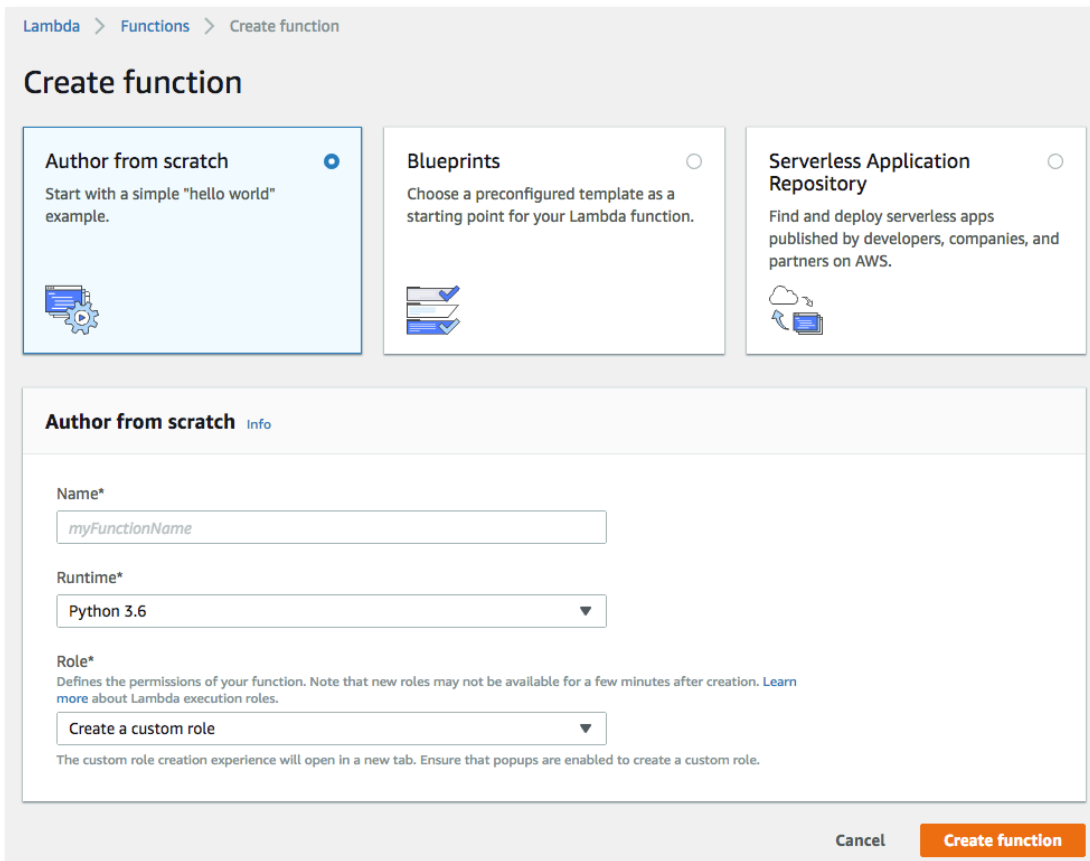


Figure 16. Lambda 함수 생성 화면.

3. Lambda 생성화면에서 Figure 16 과 같이 Lambda 함수 이름과 Runtime (Python 3.6) 그리고 Role 은 "Create a custom role"을 선택합니다.
 - a. Name : MySeq2SeqInference 을 지정.
 - b. Create a custom role 을 선택하면 Figure 17 와 같이 [AWS Lambda required access to your resources]가 나옵니다. 여기서 [Allow]를 누릅니다.
 - c. Allow 클릭하면 창이 닫히고 Lambda Console 로 돌아가는 데 여기서 Create Function 을 선택하시면 됩니다.

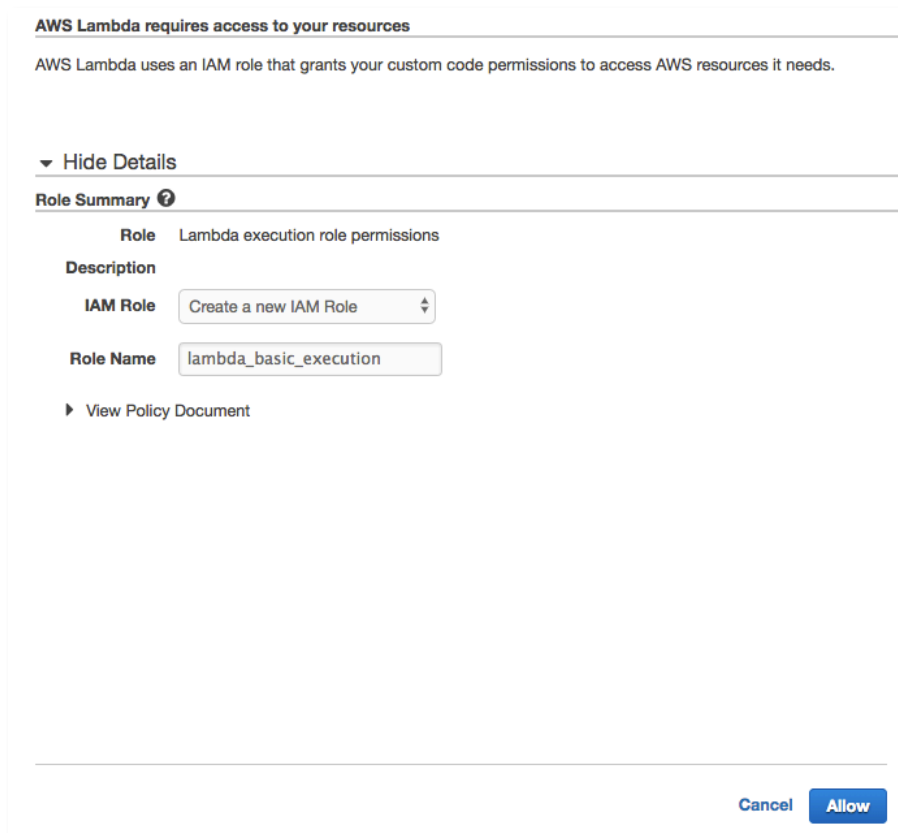


Figure 17. AWS Lambda 접근 허락 화면.

Lambda 함수에 Role 을 추가하기

방금 생성한 Lambda 함수에 새롭게 추가된 Role 에 SageMaker 와 API Gateway 를 사용할 수 있는 정책 (Policy)를 추가해보겠습니다.

1. AWS 콘솔에서 IAM 서비스를 선택하세요.
2. 왼쪽의 메뉴에서 "Roles"를 클릭하세요.

3. 방금 생성하신 Lambda 에 사용되는 Role 을 선택하세요 (Figure 18 참조)

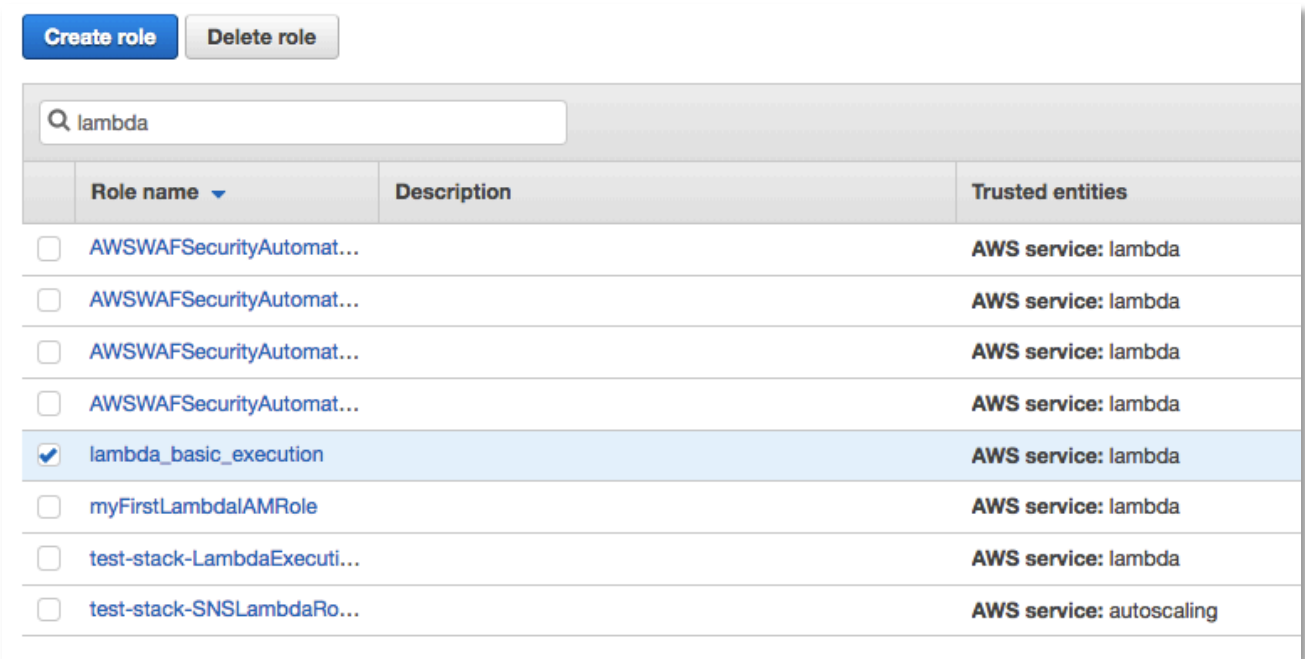


Figure 18. Lambda 함수 선택.

4. “Add inline policy”를 선택하세요 (Figure 19 참조).

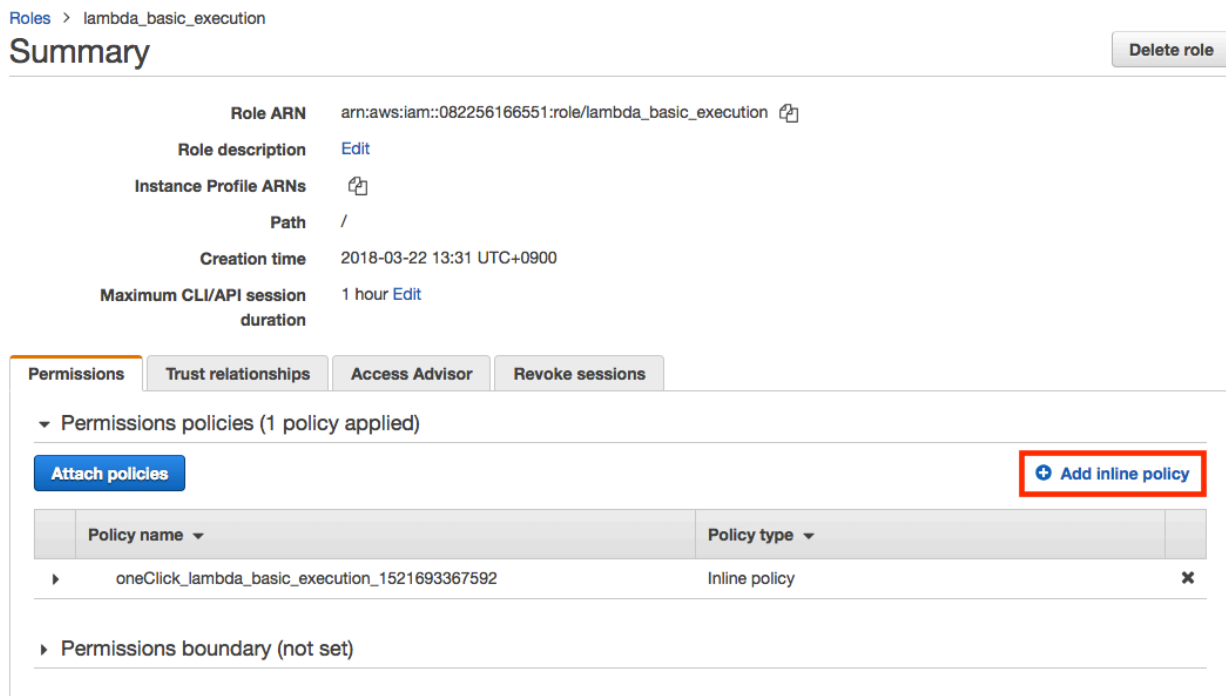


Figure 19. IAM Role 에 정책을 추가하는 화면.

5. 다음 화면의 검색창에 "SageMaker" 입력 하세요 (Figure 20 참조).

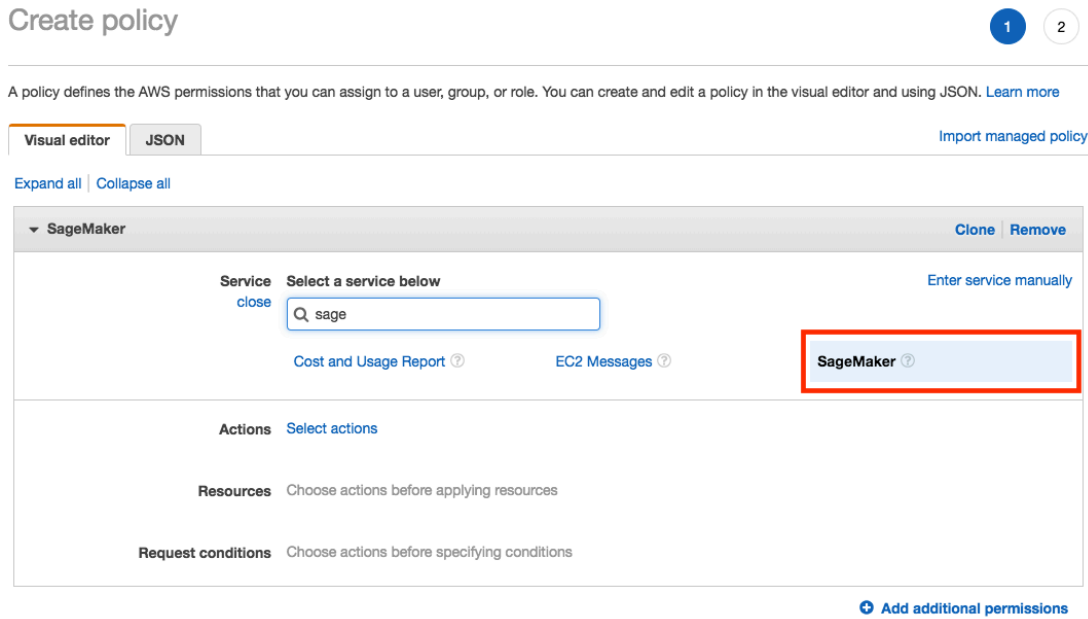


Figure 20. AmazonSageMakerFullAccess 정책 추가 화면.

6. Access level at Actions 에 있는 모든 "DescribeEndpoint" and "InvokeEndpoint" 를 선택하세요 (See Figure 21).

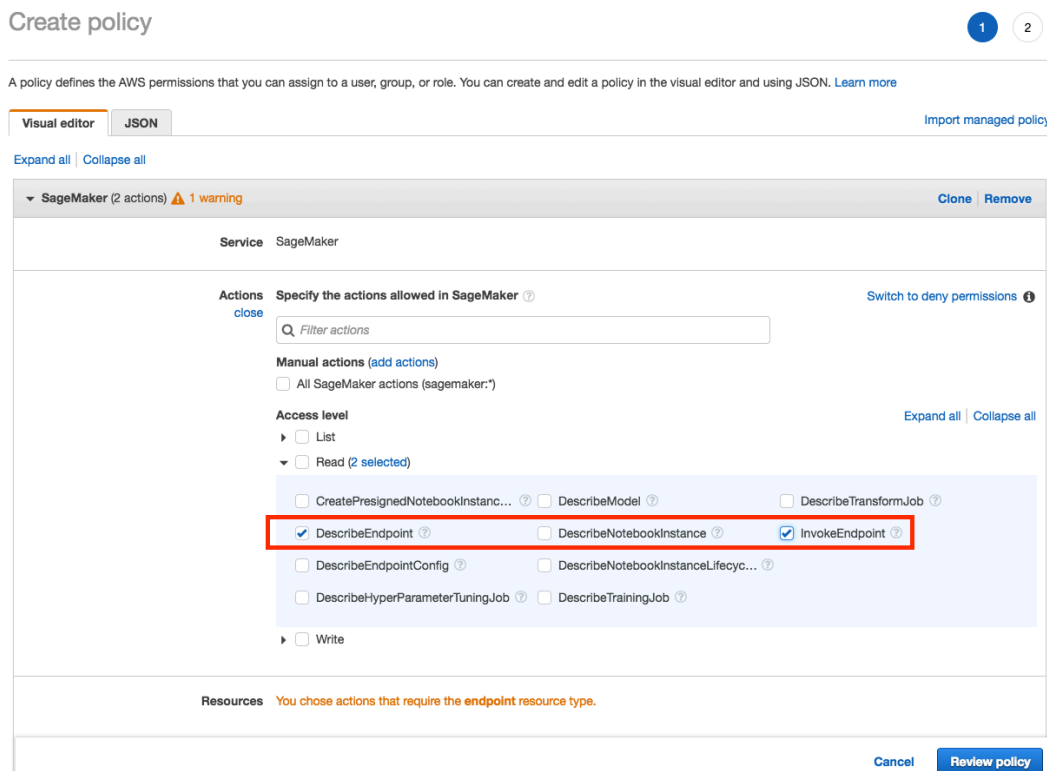


Figure 21. Select DescribeEndpoint and InvokeEndpoint in the Access level.

7. 하면 하단의 Resources 에 있는 노란색의 **“You chose actions that require the endpoint-config resource type”** 문장을 선택하신 후 Figure 22 화면과 같이 Resources 섹션에 있는 **“Any”** 를 선택합니다. 이후 화면 하단에 있는 **“Review policy”**를 선택하세요.

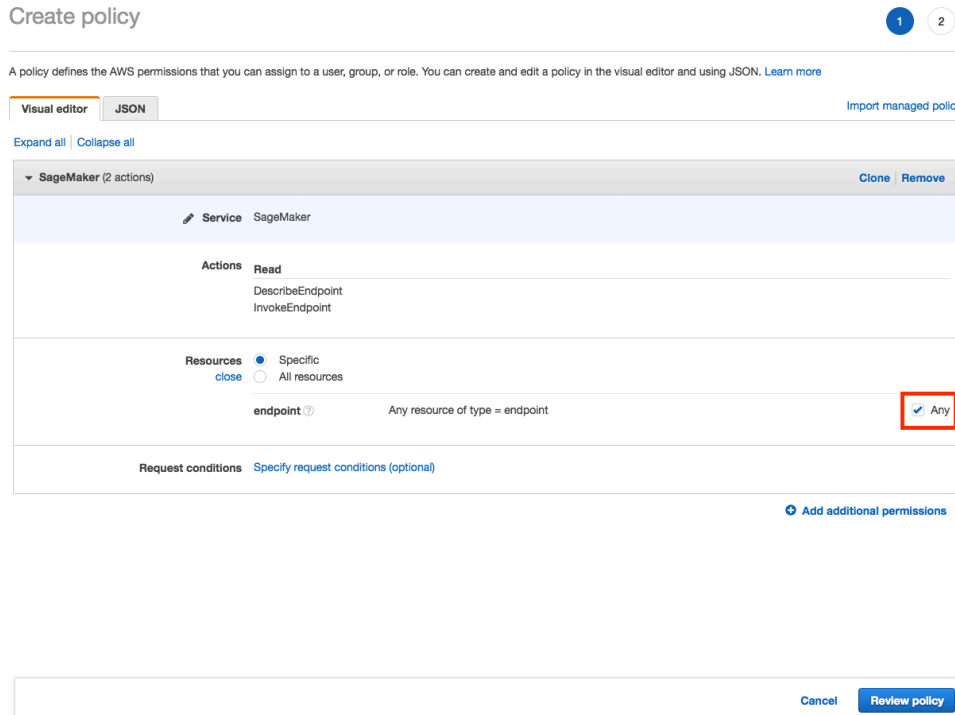


Figure 22. Select endpoint resource type.

8. “Review policy” 다이얼로그에서 새로운 policy 이름을 입력하신 후 화면 하단의 Create policy 버튼을 선택하세요 (See Figure 30).

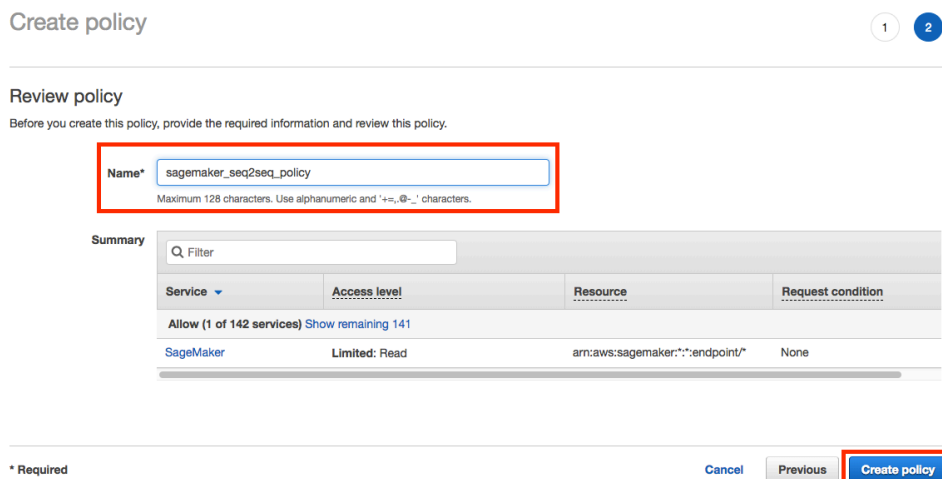


Figure 23. Create policy screen.

9. 최종 추가된 Policy 가 그림 19 와 동일한지 확인

Roles > lambda_basic_execution Delete role

Summary

Role ARN `arn:aws:iam::082256166551:role/lambda_basic_execution` [↗](#)
Role description [Edit](#)
Instance Profile ARNs [↗](#)
Path /
Creation time 2018-03-22 13:31 UTC+0900
Maximum CLI/API session duration 1 hour [Edit](#)

[Permissions](#) | [Trust relationships](#) | [Access Advisor](#) | [Revoke sessions](#)

▼ Permissions policies (2 policies applied)

[Attach policies](#) [Add inline policy](#)

Policy name ▼	Policy type ▼	
▶ oneClick_lambda_basic_execution_1521693367592	Inline policy	✕
▶ sagemaker_seq2seq_policy	Inline policy	✕

▶ Permissions boundary (not set)

Figure 24. 최종 Role 의 정책들 화면.

Lambda 함수 코딩하기

다시 AWS 콘솔의 Lambda 서비스 화면으로 이동하신 후 윗 단계에서 생성하신 Lambda 를 선택합니다. Figure 25 과 같이 추가된 Role 의 Policy 들을 확인하실 수 있습니다.

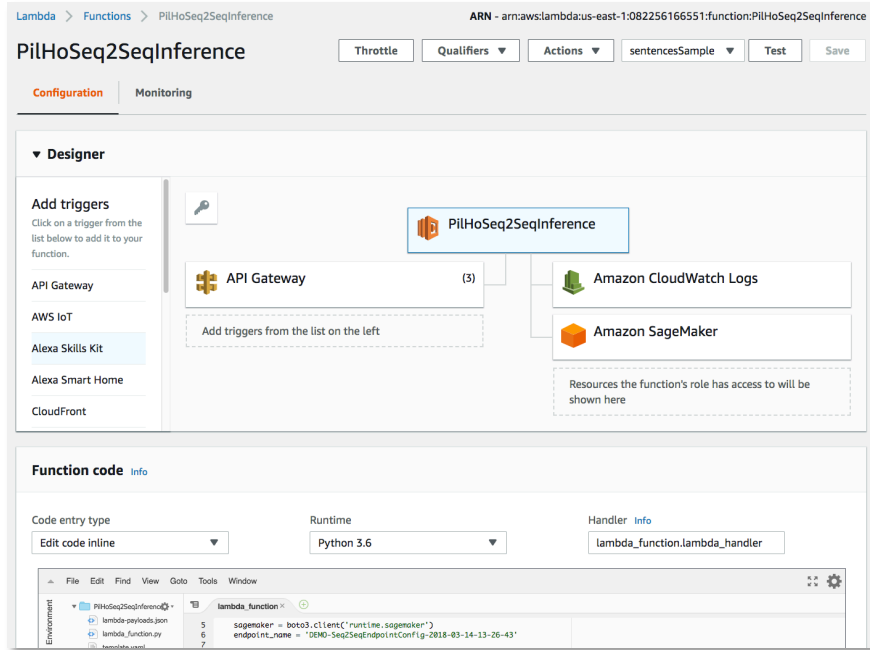


Figure 25. Lambda 선택 화면.

현 페이지에서 마우스를 스크롤해서 하단으로 이동하면 Figure 26 와 같이 Lambda 의 내장 코드들을 직접 수정할 수 있는 인터페이스가 제공이 됩니다.

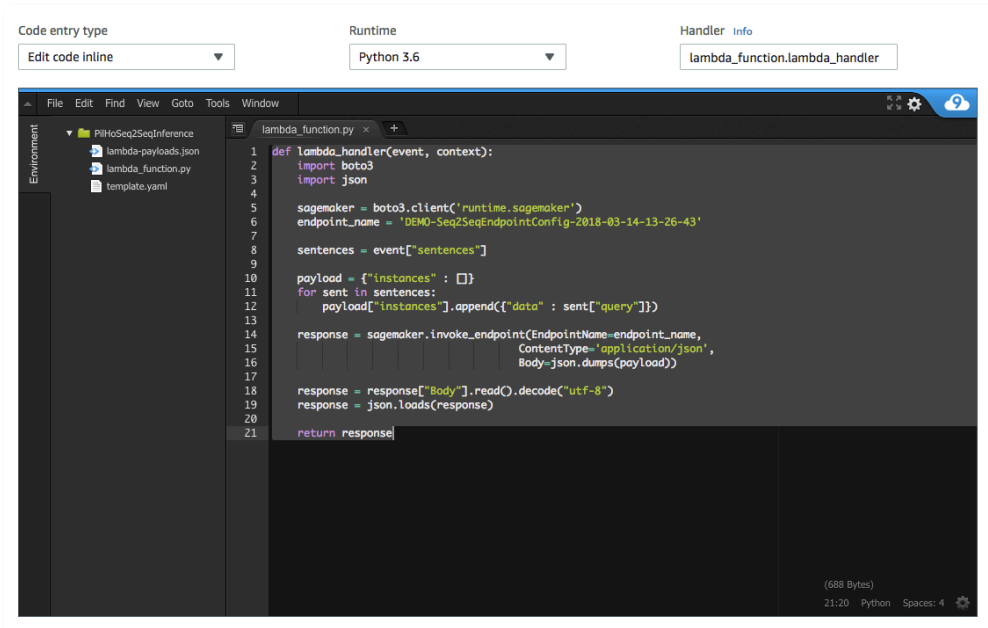


Figure 26. Lamba 코드 개발 화면.



AWS Lambda 는 AWS 콘솔 상에서 바로 코딩할 수 있게 Cloud9 에디터가 내장되어 있습니다. 아래의 순서에 따라 Lambda 함수를 만들어 보겠습니다.

1. 다음 페이지의 Python 샘플 코드를 Copy 후 Paste 로 Lambda 의 online editor 에 입력합니다. Python 코드를 복사 및 붙여 넣기를 할때는 원 코드의 indent 를 그대로 지키는 것이 중요합니다. 현재 보시고 있는 PDF 문서 상에서 복사가 제대로 되지 않는 경우 아래 온라인 주소에서 소스코드를 복사하셔도 됩니다:

https://raw.githubusercontent.com/pilhokim/ai-ml-workshop/master/2018-09/lambda_function.py

2. 붙여넣기 하신 소스코드 상의 "endpoint_name" 을 본 실습 동안 생성한 Seq2Seq endpoint 서버 주소로 변경하십시오 (Figure 27 참조).

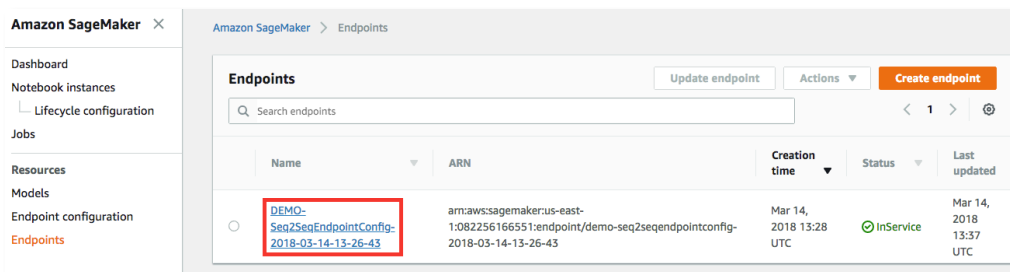


Figure 27. SageMaker EndPoint 이름 확인 방법.

Labmda Python sample Code

```
def lambda_handler(event, context):
    import boto3
    import json

    sagemaker = boto3.client('runtime.sagemaker')
    endpoint_name = 'YourSeq2SeqEndpointName'

    sentences = event["sentences"]

    payload = {"instances" : []}
    for sent in sentences:
        payload["instances"].append({"data" : sent["query"]})

    response = sagemaker.invoke_endpoint(EndpointName=endpoint_name,
                                         ContentType='application/json',
                                         Body=json.dumps(payload))

    response = response["Body"].read().decode("utf-8")
    response = json.loads(response)

    return response
```

- Endpoint 용으로 선택하신 서버의 Instance Type 과 번역을 하기위한 text 의 크기에 따라 번역에 몇초 이상이 소요될 수도 있습니다. 이 시간동안 Lambda 함수 호출이 Timeout 되는 것을 방지하기 위해 Figure 28와 같이 Lambda의 Timeout 시간을 10 초로 늘입니다.
- 상단의 "Save" 버튼을 눌러 저장합니다.

The screenshot shows the 'Basic settings' section of the AWS Lambda console. It includes a 'Description' text area, a 'Memory (MB)' slider set to 128 MB, and a 'Timeout' field. The 'Timeout' field is set to 0 minutes and 10 seconds, with the '10' and 'sec' parts highlighted by a red box.

Figure 28. Lambda 함수 Timeout 값 조정.

새로 만든 Lambda 함수의 동작을 바로 확인할 수 있습니다.

- Figure 29와 같이 "Configure test events"를 선택합니다.

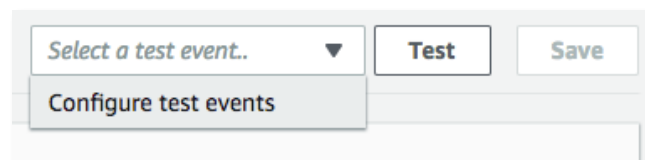


Figure 29. Lambda 테스트 데이터 구성 화면.

- 테스트 이벤트 입력화면에서 Figure 30과 같이 아래의 샘플 영어 문장을 입력합니다. 또는 https://raw.githubusercontent.com/pilhokim/ai-ml-workshop/master/2018-09/sample_query.json 에서 복사해서 사용하셔도 됩니다.

```
{
  "sentences": [
    {
      "query": "I love you"
    },
    {
      "query": "I love you, too"
    }
  ]
}
```

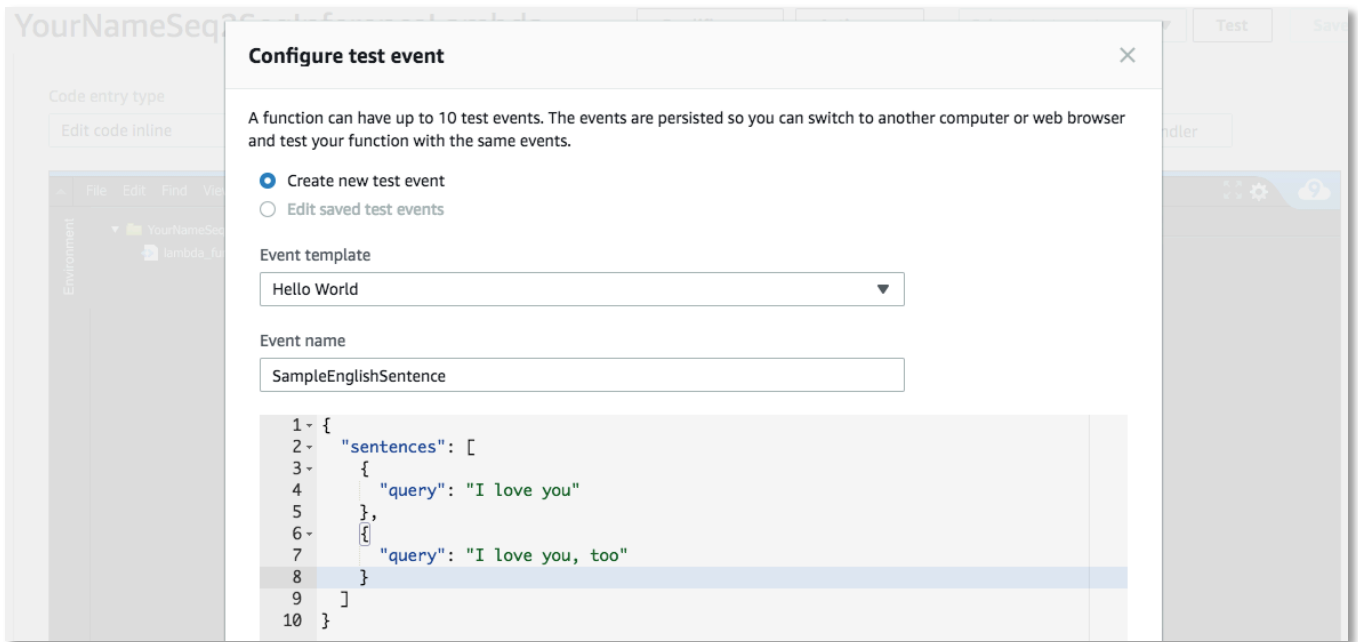



Figure 30. Test 이벤트 생성.

이때 주의하실점은 JSON 형식의 "sentences"와 "query"는 미리 약속된 key 값이므로 변경을 하시면 안됩니다.

3. 입력이 완료 된 후 상단의 "Test" 버튼을 클릭하시면 Figure 31 와 같은 화면이 보이면 정상적으로 작동하는 것을 확인하실 수 있습니다. 하단의 Cloud9 에서도 결과를 확인하실 수 있습니다.

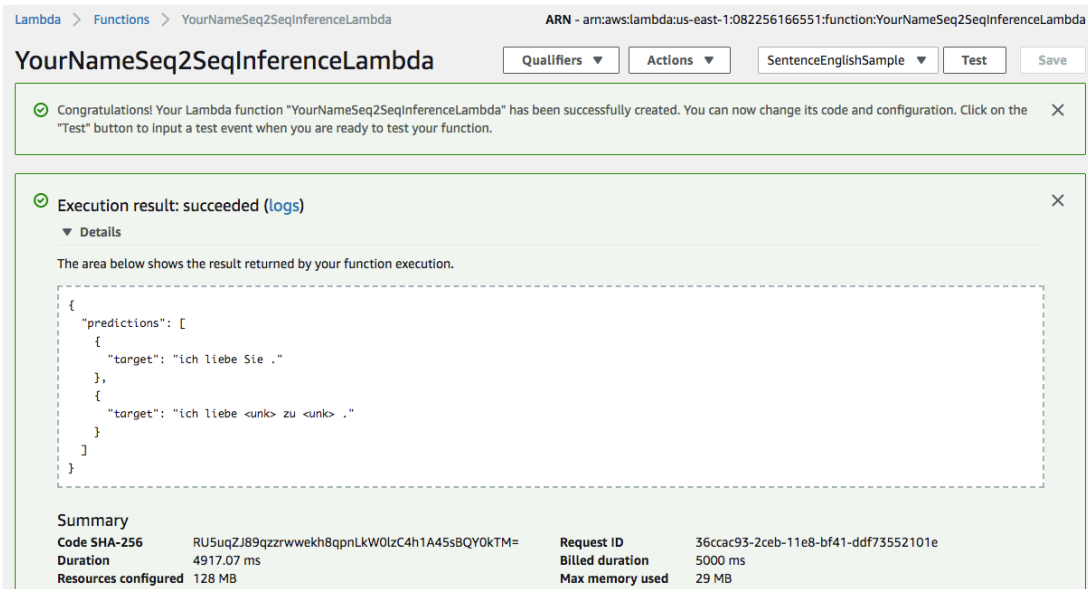


Figure 31. Lambda 함수 테스트 결과 화면.

Module 8-3: AWS API Gateway 와 S3 Static Web Server 를 이용한 웹서비스 연결하기

API Gateway 생성 및 Lambda 함수 연결하기

1. Amazon API Gateway 콘솔 접속 (<https://console.aws.amazon.com/apigateway/>)
2. "Create API" -> "New API" 선택
3. 셋팅에서 새로운 API name 입력 (ex. SageMakerSeq2SeqLambdaGateWay)후 Endpoint Type 을 Regional 로 선택 (Figure 32 참조).

Create new API

In Amazon API Gateway, an API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

New API
 Clone from existing API
 Import from Swagger
 Example API

Settings

Choose a friendly name and description for your API.

API name* SageMakerSeq2SeqLambdaGateWay
Description
Endpoint Type Regional

* Required Create API

Figure 32. Amazon API Gateway 생성 화면.

4. 바뀐 화면에서 Actions -> Create Method 선택
5. 하단의 콤보 박스에서 POST 선택 (Figure 33 참조)
6. 체크(V) 버튼 클릭해서 적용 (Figure 33 참조)

Resources **Actions** / **Methods**

/

POST

Figure 33. POST method 추가 화면.

7. 오른쪽의 셋업에서 아래와 같이 입력 진행 (Figure 34 참조)
 - a. Integration type: Lambda function
 - b. Lambda region: Lambda 를 생성하신 Region (**us-east-1**) 입력
 - c. Lambda function: Lambda 함수 이름 입력
 - d. "Save" 선택

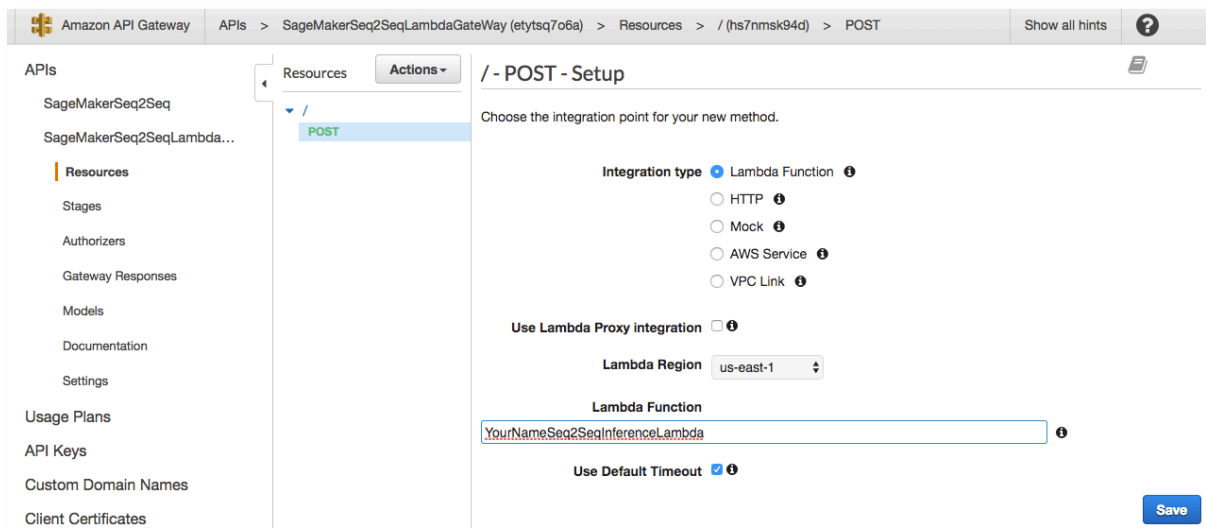


Figure 34. Lambda 함수를 호출하기 위한 Gateway POST method 셋팅 화면.

API Gateway 가 생성이 된 이후에는 Figure 35 와 같이 Test 를 진행하여 제대로 Lambda 를 호출하는지 확인하실 수 있습니다.

- a. "Test"를 선택하셔서 API Gateway 의 testing interface 를 확인합니다.
- b. Request body 에 Lambda 호출에 사용되었던 아래의 예제 데이터를 입력하신 후 Test 를 선택합니다.

```
{
  "sentences": [
    {
      "query": "I love you"
    },
    {
      "query": "I love you, too"
    }
  ]
}
```

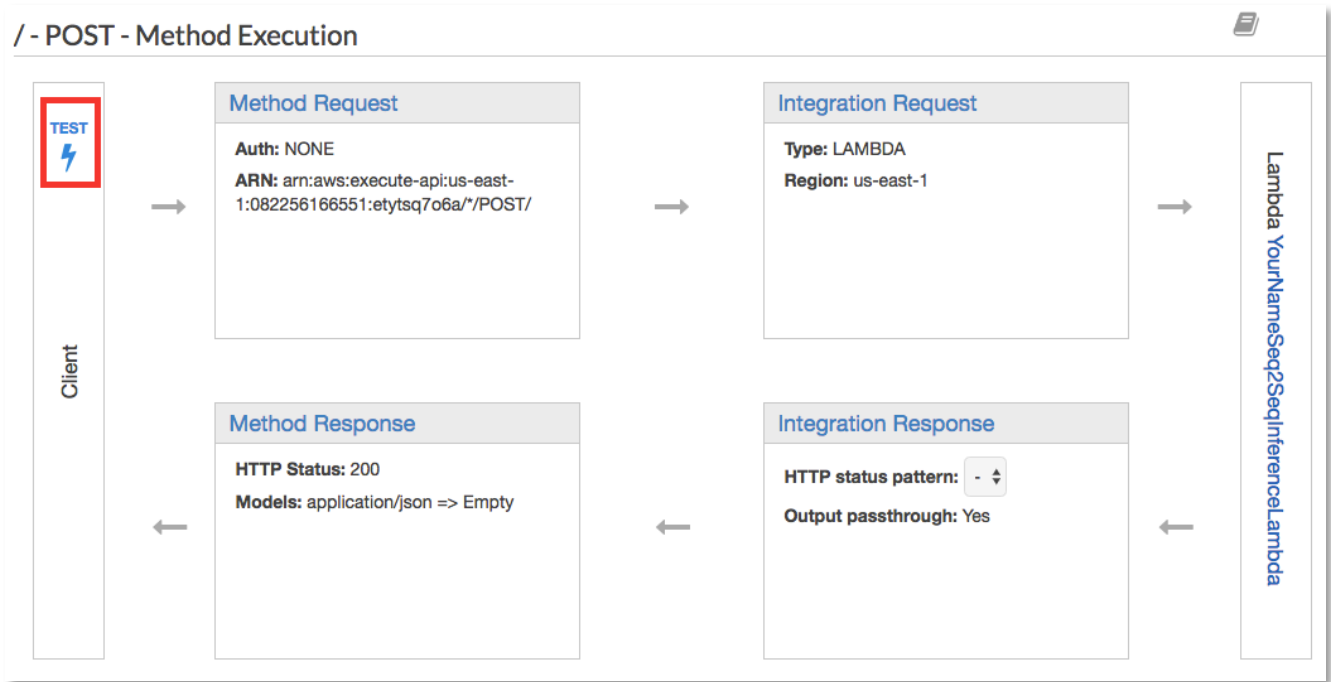


Figure 35. API Gateway Test 화면

테스트 결과가 Figure 36 과 같이 보이면 정상적으로 동작하는 것으로 확인하실 수 있습니다.

The screenshot shows the 'Method Execution' page for a POST method. The left sidebar lists various API Gateway components, with 'Resources' expanded to show the selected resource. The main content area displays the following information:

- Path:** No path parameters exist for this resource. You can define path parameters by using the syntax `{myPathParam}` in a resource path.
- Query Strings:** No query string parameters exist for this method. You can add them via Method Request.
- Headers:** No header parameters exist for this method. You can add them via Method Request.
- Stage Variables:** No stage variables exist for this method.
- Request Body:**

```

1 - {
2 -   "sentences": [
3 -     {
4 -       "query": "I love you"
5 -     },
6 -     {
7 -       "query": "I love you, too"
8 -     }
9 -   ]
10 - }
11 -
            
```
- Request Summary:**
 - Request: /
 - Status: 200
 - Latency: 5812 ms
 - Response Body:


```

{
  "predictions": [
    {
      "target": "ich liebe Sie ."
    },
    {
      "target": "ich liebe <unk> zu <unk> ."
    }
  ]
}
                    
```
- Response Headers:**

```

{"X-Amzn-Trace-Id":"sampled=0;root=1-5ab304a4-7bbb9e9a517c1be889c97374","Content-Type":"application/json"}
            
```
- Logs:**

```

Execution log for request test-request
Thu Mar 22 01:19:32 UTC 2018 : Starting execution for request: test-invoke-request
Thu Mar 22 01:19:32 UTC 2018 : HTTP Method: POST, Resource Path: /
Thu Mar 22 01:19:32 UTC 2018 : Method request path: {}
            
```

Figure 36. API Gateway 테스트 결과.

8. Enable CORS: S3 Static Web Server 를 이용해서 API Gateway 를 호출하면 origin 이 다르기 때문에 반드시 CORS (Cross-Origin Resource Sharing)를 Enable 해야만 외부 사이트에서 이 REST 서비스를 이용할 수 있게 됩니다.

a. Actions -> Enable CORS 선택 (Figure 37 참조)

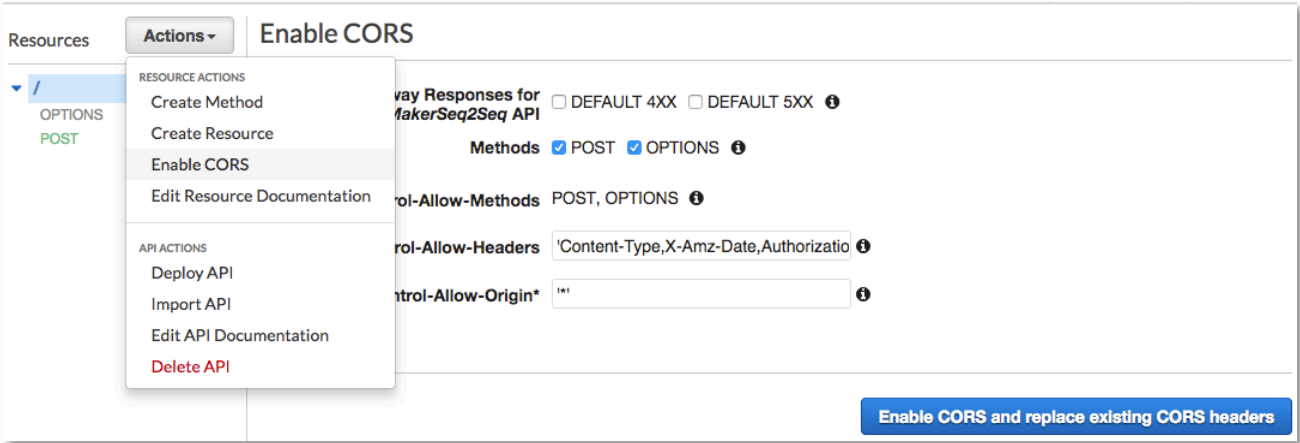


Figure 37. API Gateway API Enable CORS 화면.

b. "Enable CORS and replace existing CORS headers" 선택

c. "Yes, replace existing values" 선택 (Figure 38 참조)

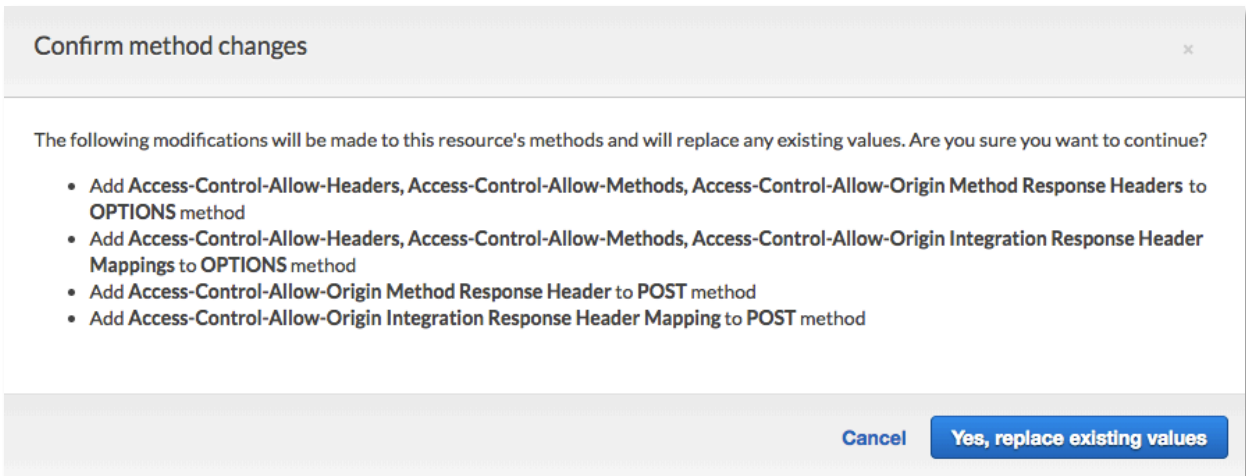


Figure 38. CORS replace existing values 화면.

9. 정상적으로 동작이 되면 Actions->Deploy API 선택 (Figure 39 참조) 합니다. API Deploy 를 반드시 하셔야 실제 외부 (Public Internet)에서 호출을 할 수 있습니다.

10. 현재 생성한 Gateway 의 stage 이름을 부여합니다. 예제에서는 "prod"라는 약어로 stage 이름을 정의하였습니다. 개발 단계에 따라 "test" 나 "prod" 등 의미 있는 키워드를 부여하시면 됩니다.

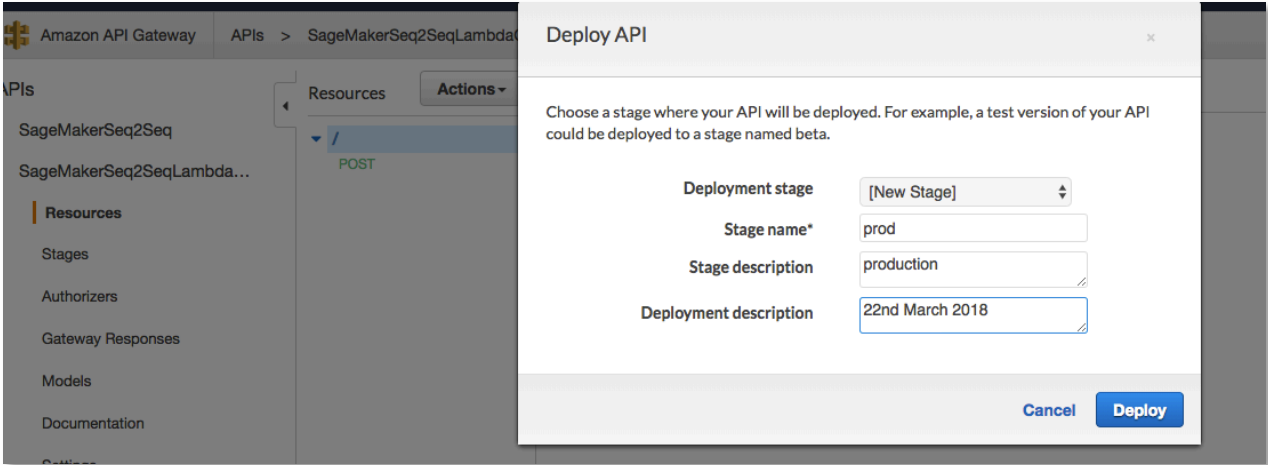


Figure 39. API deploy 화면.

11. Deploy 가 된 이후 Stage Editor 에서 invoke URL 을 (Figure 40 참조) 메모장에 따로 기록해 두시고 "SDK Generation" -> Platform (JavaScript) -> Generate SDK 선택. 이 JavaScript 라이브러리는 API Gateway 서비스에 대해 [CORS](#) (Cross-Origin Resource Sharing)을 지원해주는 기능을 포함하고 있습니다.

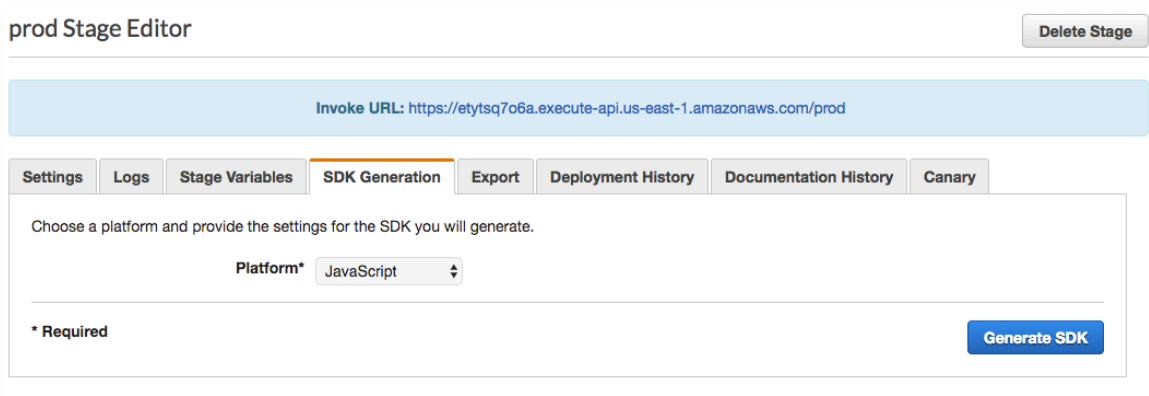


Figure 40. API Gateway 접속 SDK 다운로드 화면

이제 S3 를 이용해서 static web server 를 설정하기 위한 파일들을 준비하겠습니다.

- a. 상기 API Gateway SDK 생성으로 다운 받은 압축 파일을 임의의 디렉토리에 푸세요 (unzip).



- b. S3 Static 웹 서버에 사용될 index.html 과 error.html 파일을 다음의 S3 버킷에서 다운로드 하여 상기 단계에서 사용된 디렉토리에 동일하게 저장합니다:
https://s3.amazonaws.com/pilho-sagemaker-ai-workshop-lambda/index_error_html.zip
- c. 최종 파일들이 Figure 41 과 같이 구성되어 있으면 됩니다. 이 파일들은 다음 단계에서 만들 S3 버킷에 업로드 되게 됩니다.

Name	Date Modified	Size	Kind
lib	Today, 10:56 AM	--	Folder
apigClient.js	Today, 1:56 AM	4 KB	JavaScript
error.html	Today, 11:04 AM	53 bytes	HTML
index.html	Today, 11:04 AM	2 KB	HTML
README.md	Today, 1:56 AM	3 KB	Markdown

Figure 41. 웹서버 구성 파일 리스트 화면.

S3 Static Web Server 생성하기

1. Amazon S3 콘솔 접속 (<https://s3.console.aws.amazon.com>)
2. "Create bucket" 선택
3. 새로운 버킷 이름 입력 (ex. "jihye-sagemaker-public-test") -> Next -> Next 선택
4. Set permissions 에서 Manage public permissions 를 "Grant public read access to this bucket" 으로 설정 (Figure 43 참조)

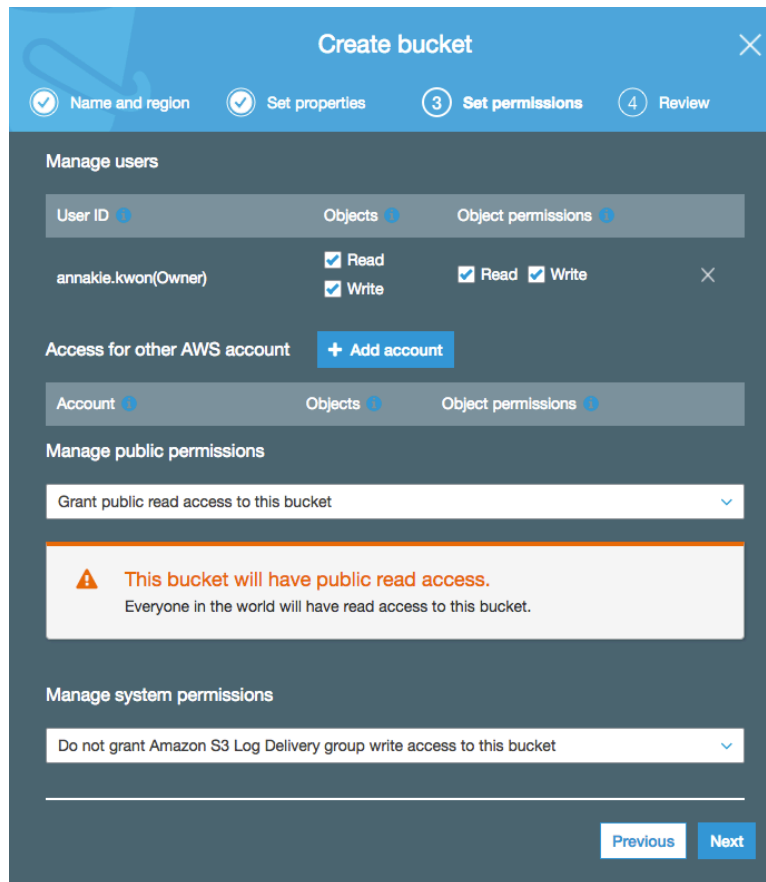


Figure 42. S3 버킷 Public 접속 허용 화면.

5. Next->Create bucket 선택
6. 생성된 S3 bucket 선택
7. "Properties" -> "Static website hosting" -> "Use this bucket to host a website" 선택 후 Index document : index.html, Error document : error.html 입력
8. "Save" 선택 (Figure 43 참조)
9. 이 단계 까지 마치신 후 상단의 URL 형식의 Endpoint 주소를 기록해 둡니다. 이 URL 주소를 이용해서 S3 웹 서버에 접속하게 됩니다.

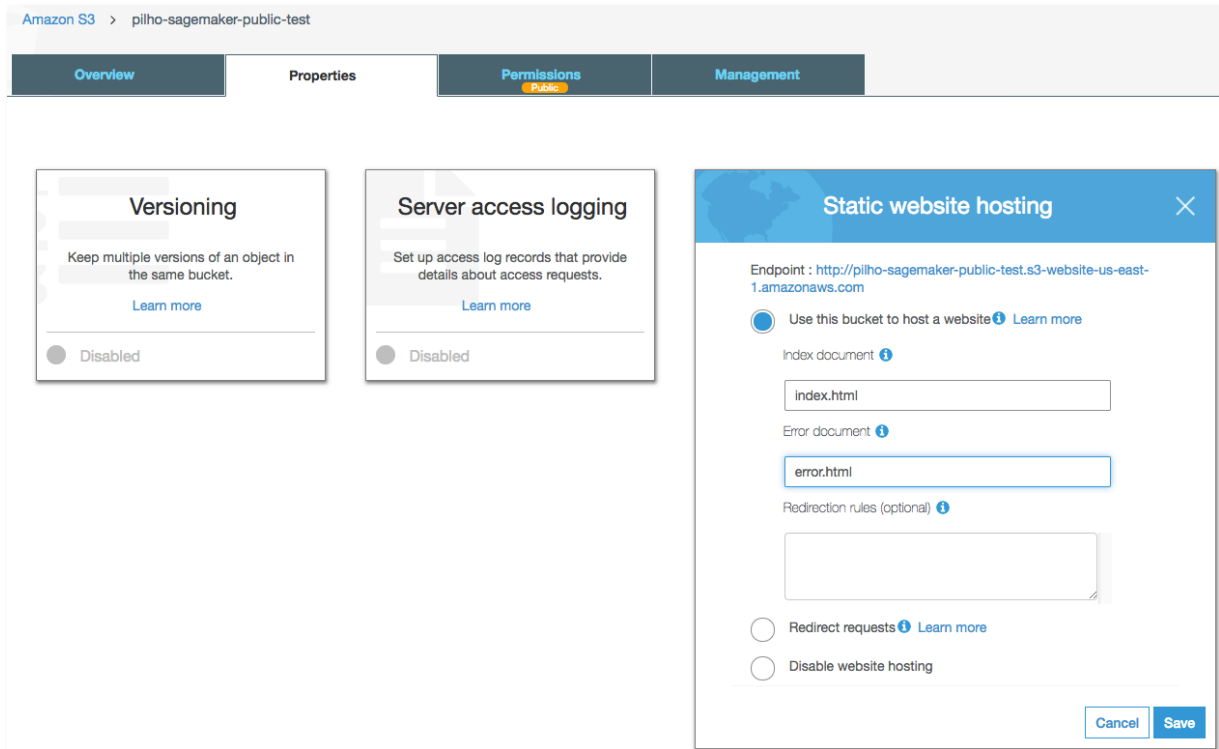


Figure 43. S3 static 웹서버 설정 화면.

10. "Overview" 탭 선택 -> "Upload" 선택

11. 생성된 S3 Bucket 에 이전 단계에서 생성된 파일들을 업로드. 이때 Set permissions 을 반드시 Grant public read access to this object(s)로 설정해야 합니다.

최종 서비스 테스트하기

1. 웹 브라우저를 구동하시고 S3 Endpoint URL 에 접속합니다 (Figure 44 참조)
2. Translate to German 오른쪽의 텍스트 입력 창에 영문 문장을 입력합니다. (Ex. "I love you")
3. 몇 초 정도 기다리시면 하단에 번역 결과가 보여집니다.

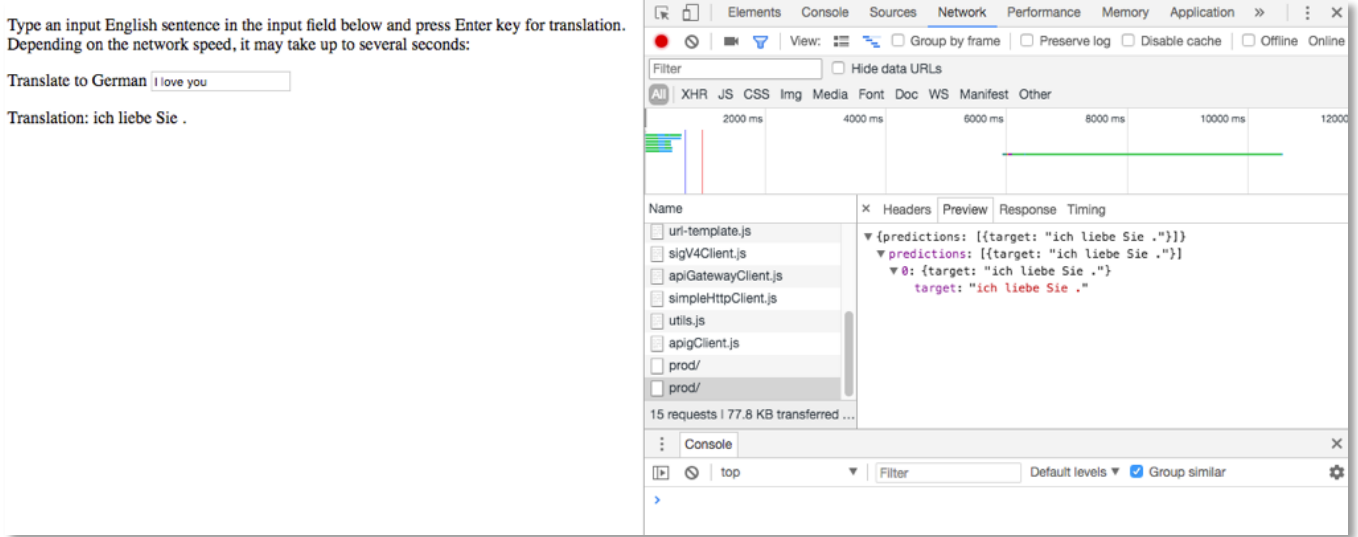


Figure 44. 웹기반 번역 서비스 테스트 화면.

SageMaker Endpoint 서버 자동 확장 설정하기

본 섹션은 향후 실제 필요시에 대한 참조용으로 제공됩니다. 실제 Hands-on 을 하실 필요는 없습니다.

웹 기반 서비스를 제공하기 시작하고 사용자 수가 증가하기 시작하면 SageMaker 의 Inference 서버도 자동으로 확장되게 설정하실 수 있습니다.

```
In [12]: from time import gmtime, strftime

endpoint_config_name = 'Seq2SeqEndpointConfig-' + strftime("%Y-%m-%d-%H-%M-%S", gmtime())
print(endpoint_config_name)
create_endpoint_config_response = sage.create_endpoint_config(
    EndpointConfigName = endpoint_config_name,
    ProductionVariants=[{
        'InstanceType': 'ml.m4.xlarge',
        'InitialInstanceCount': 1,
        'ModelName': model_name,
        'VariantName': 'AllTraffic' }])

print("Endpoint Config Arn: " + create_endpoint_config_response['EndpointConfigArn'])

Seq2SeqEndpointConfig-2018-03-24-08-35-50
Endpoint Config Arn: arn:aws:sagemaker:us-east-1:082256166551:endpoint-config/seq2seqendpoint
config-2018-03-24-08-35-50
```

Figure 45. Endpoint 설정에서 InitialInstanceCount 변수 화면.

Figure 45 와 같이 Endpoint 서버 설정에서의 Instance count 는 "InitialInstanceCount"로 설정이 됩니다. 즉 초기의 서버 갯수 만을 설정하는 것이고 사용자의 요청 부하에 따라 서버 설정이 바뀌게 할 수 있습니다. 아래에는 AWS SageMaker 콘솔을 이용해서 autoscaling 을 설정 하는 방법을 보겠습니다.



1. AWS SageMaker 콘솔에서 왼편의 **Endpoints** 를 선택하신 후 오른편 화면에서 생성하신 Endpoint 를 선택합니다 (Figure 46 참조).

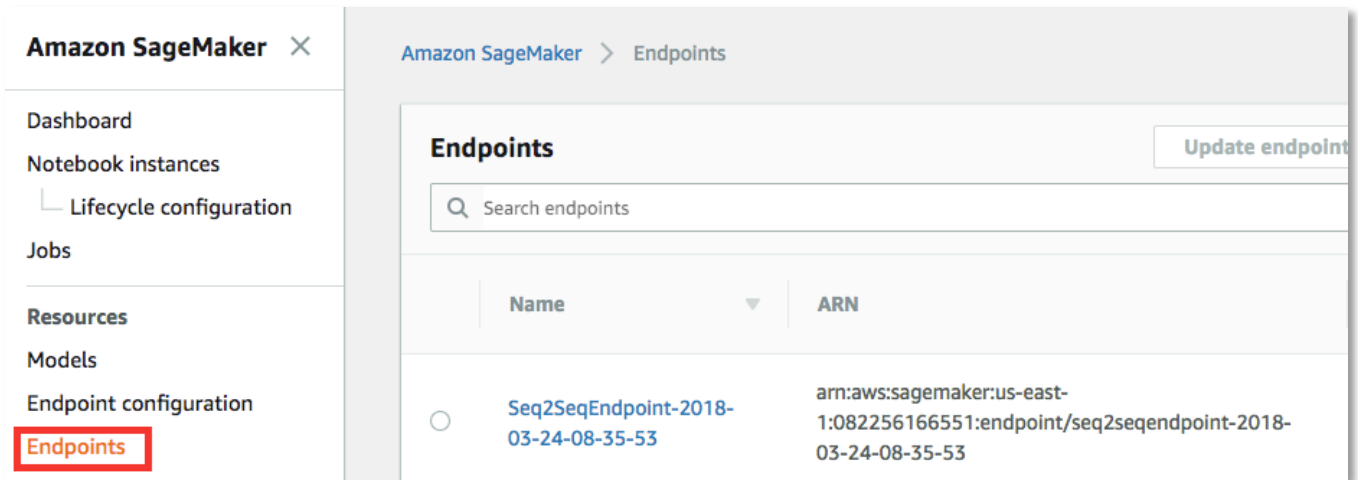


Figure 46. AWS 콘솔에서 SageMaker 의 Endpoints 선택 화면

2. 선택된 Endpoint 내용 화면에서 스크롤을 하셔서 **Endpoint runtime settings** 에서 **AllTraffic** 을 선택하신 후 오른편의 **Configure auto scaling** 버튼을 선택합니다. 참고로 이 화면에서 각 Variant 별 Weight 변경 (**Update Weights**)와 평상시 서버 개수 (**Update Instance count**) 도 변경하실 수 있습니다.

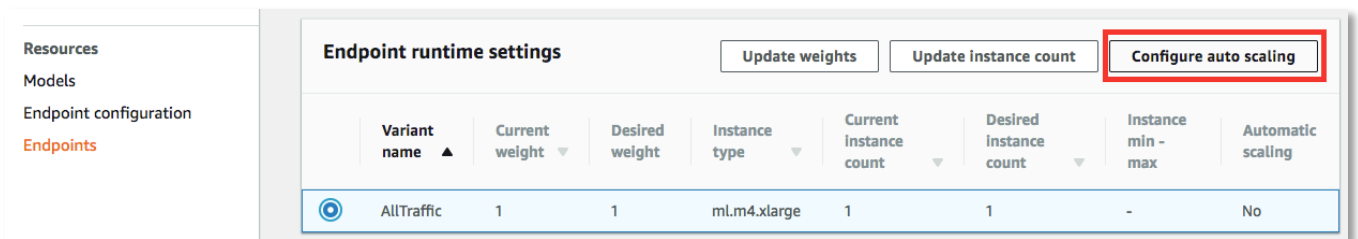


Figure 47. Auto scaling 설정 화면.

3. Configure variant automatic scaling 화면에서는 Variant automatic scaling 과 Scaling policy 를 설정 하실 수 있습니다 ([참조링크](#)). Amazon SageMaker 는 [target-tracking scaling 정책](#)을 사용하고 있습니다. 즉 미리 정의된 metric 이나 custom metric 을 사용하셔서 target value 를 지정하실 수 있는데 CloudWatch 알람을 통해 scaling 정책을 구동 시키고 instance server scale 을 조정하실 수 있습니다. 본 핸드온에서는 직접 다루지는 않지만 [참조링크](#)를 통해 좀 더 자세한 내용을 파악해 보시는 것도 좋을 것 같습니다.



Configure variant automatic scaling

[Deregister auto scaling](#)

Variant automatic scaling [Learn more](#)

Variant name	Instance type	Current instance count	Current weight
AllTraffic	ml.m4.xlarge	1	1

Minimum instance count: - Maximum instance count:

IAM role
 Amazon SageMaker uses the following service-linked role for automatic scaling. [Learn more](#)
 AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint

Built-in scaling policy [Learn more](#)

Policy name: SageMakerEndpointInvocationScalingPolicy

Target metric	Target value
SageMakerVariantInvocationsPerInst	<input type="text" value="70"/>
Scale in cool down (seconds) - optional	Scale out cool down (seconds) - optional
<input type="text" value="300"/>	<input type="text" value="300"/>

Disable scale in
 Select if you don't want automatic scaling to delete instances when traffic decreases. [Learn more](#)

Figure 48. Automatic scaling 정책 설정 화면.

이상으로 모듈 6의 실습 과정을 마무리 하셨습니다. 워크샵 이후 발생하는 비용을 방지하기 위해 다음 페이지의 서비스 종료 가이드를 통해 사용하신 리소스들을 모두 종료/삭제 해주십시오.

서비스 종료 가이드

워크 샵 이후 발생 되는 비용을 방지하기 위해서 아래의 단계에 따라 모두 종료/삭제 해 주세요. 비용이 발생하더라도 실습하신 Internet-facing App 을 유지하고 싶으신 경우에는 아래의 Notebook instance 의 경우만 처리하시면 됩니다.

- **Notebook instance:**

1) 만약 향후 사용을 위해 인스턴스를 저장하고 싶다면 **stop** 을 하시면 됩니다. 이 경우 스토리지 비용은 발생합니다. 향후 다시 재가동 하시려면 Start button 을 클릭하면 됩니다.

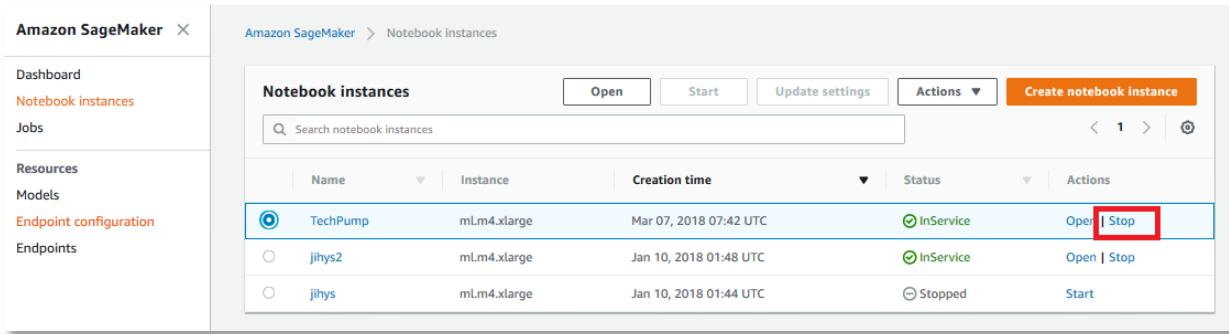


Figure 49. SageMaker 노트북 인스턴스 중단 화면.

2) 삭제를 할 경우는 **stop** 되어 있는 해당 notebook instance 를 선택하고 **Action** Dropdown 메뉴에서 **Delete** 선택 하시면 됩니다.

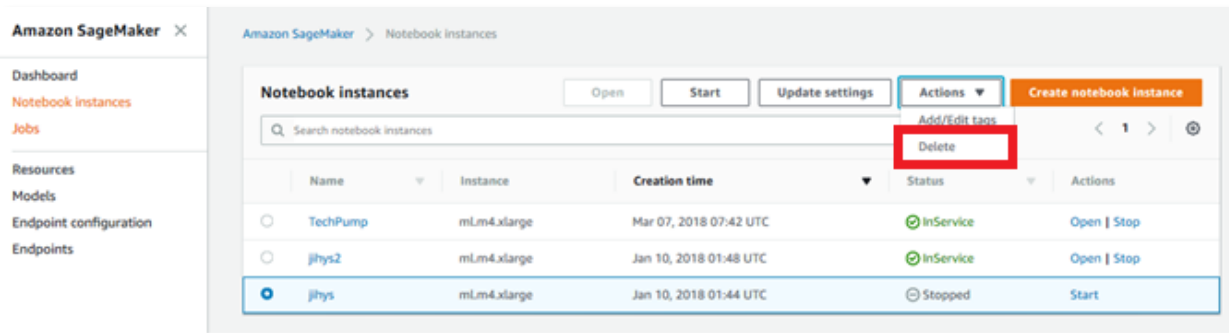


Figure 50. SageMaker 노트북 인스턴스 삭제 화면.

- **SageMaker Endpoints:**

훈련된 모델을 실제 예측 업무를 위해 배포된 한대 이상으로 구성된 클러스터입니다. Notebook 안에서 명령어로 삭제하거나 SageMaker console 에서 삭제 하실 수 있습니다. 삭제



하시기 위해서는 왼쪽 패널의 Endpoints 를 선택 하신 후 해당 endpoints 들 옆에 radio button 을 클릭 하신 후 Action Dropdown 메뉴에서 Delete 선택 하시면 됩니다.

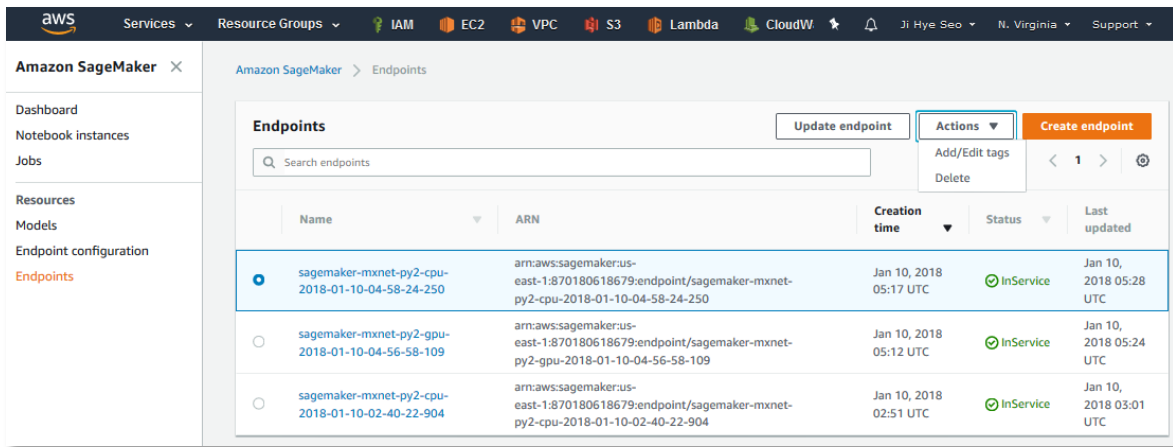


Figure 51. SageMaker Endpoint 삭제 화면.

- Lambda instance: 생성하신 Lambda instance 를 삭제합니다.

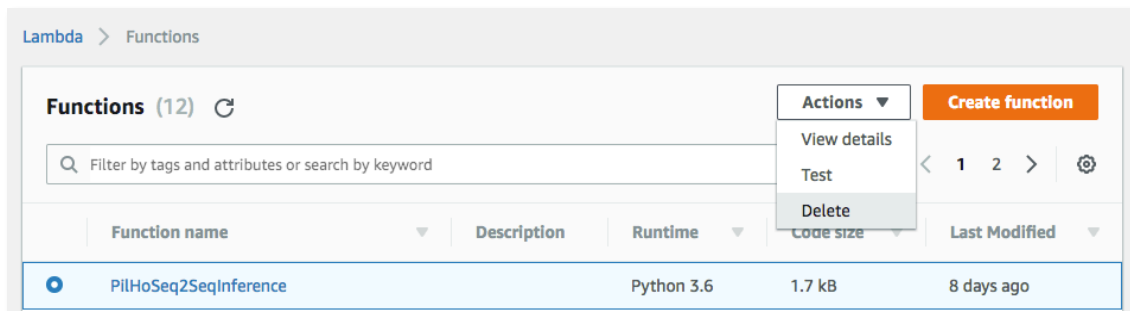


Figure 52. Lambda 인스턴스 삭제 화면.

- Amazon API Gateway instance: 생성하신 Gateway instance 를 삭제합니다.

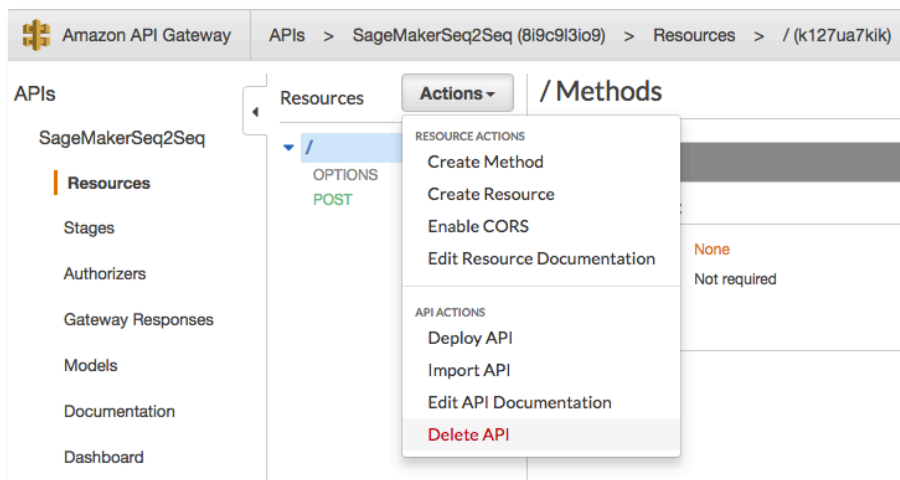


Figure 53. API Gateway 삭제 화면.

- Amazon S3 buckets: 생성하신 S3 Bucket (SageMaker 용, Public Internet 용)들을 모두 삭제합니다.

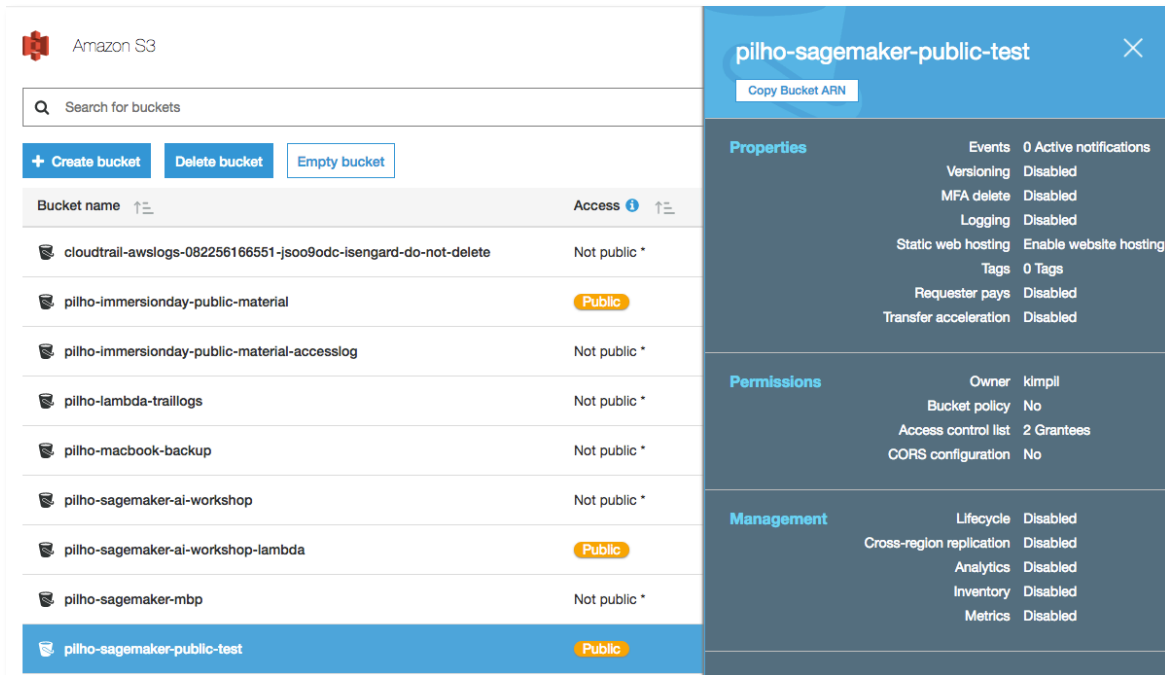


Figure 54. S3 버킷 삭제 화면.

이상으로 본 핸즈온 세션의 모든 과정을 마무리 하셨습니다. 수고하셨습니다.