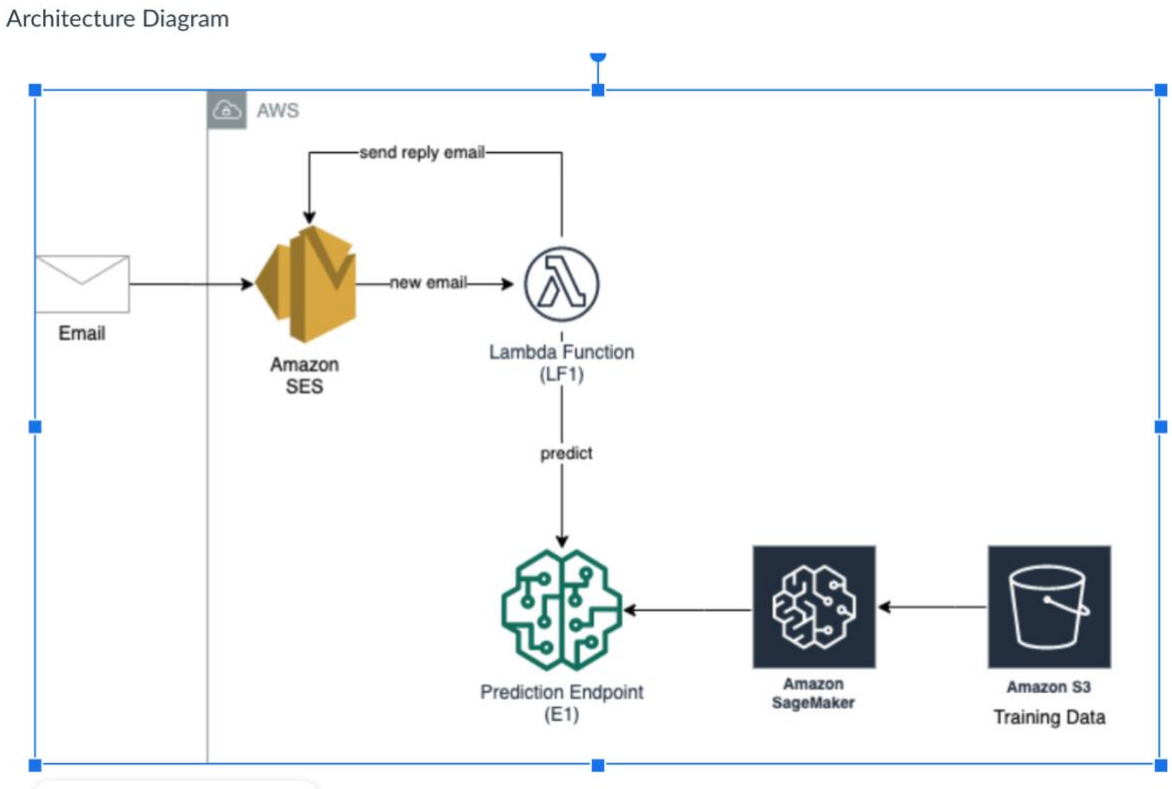
Homework Assignment 3: ML Ops::Spam Detection

Due Date: 04/19 11:59pm

In this assignment you will implement a machine learning model to predict whether a message is spam or not. Furthermore, you will create a system that upon receipt of an email message, it will automatically flag it as spam or not, based on the prediction obtained from the machine learning model.

Architecture Diagram :



Outline:

This assignment has the following components:

- **Complete tutorial for using Amazon SageMaker on AWS.**

    0. Follow the following AWS tutorial on how to use Amazon SageMaker to implement the required model: https://aws.amazon.com/getting-started/hands-

on/build-train-deploy-machine-learning-model-sagemaker/ (Links to an external site.)

1. The purpose of the tutorial is to familiarize you with Amazon Sagemaker and the basic components of SageMaker.

There is a change that is to be made due to Sagemaker updates:

Change framework_version from 1.6 to 1.2

- **Implement a Machine Learning model for predicting whether an SMS message is spam or not.**

    0. Follow the following AWS tutorial on how to build and train a spam filter machine learning model using Amazon SageMaker: https://github.com/aws-samples/reinvent2018-srv404-lambda-sagemaker/blob/master/training/README.md (Links to an external site.)
    1. The resulting model should perform well on emails as well, which is what the rest of the assignment will focus on.
    2. Deploy the resulting model to an endpoint (E1).

- **Implement an automatic spam tagging system.**

    0. Create an S3 bucket (S1) that will store email files.
    1. Using SES, set up an email address, that upon receipt of an email it stores it in S3.
        0. Confirm that the workflow is working by sending an email to that email address and seeing if the email information ends up in S3.
    2. For any new email file that is stored in S3, trigger a Lambda function (LF1) that extracts the body of the email and uses the prediction endpoint (E1) to predict if the email is spam or not.
        0. You might want to strip out new line characters "\n" in the email body, to match the data format in the SMS dataset that the ML model was trained on.
    3. Reply to the sender of the email (it could be your email, the TA's etc.) with a message as follows:

        "We received your email sent at [EMAIL_RECEIVE_DATE] with the subject [EMAIL_SUBJECT].

Here is a 240 character sample of the email body: [EMAIL_BODY]

The email was categorized as [CLASSIFICATION] with a [CLASSIFICATION_CONFIDENCE_SCORE]% confidence."

0. Replace each variable "[VAR]" with the corresponding value from the email and the prediction.
1. The purpose of this step is to facilitate easy testing.

- **Create an AWS CloudFormation template for the automatic spam tagging system.**

    0. Create a CloudFormation template (T1) to represent all the infrastructure resources (ex. Lambda, SES configuration, etc.) and permissions (IAM policies, roles, etc.).
    1. The template (T1) should take the prediction endpoint (E1) as a stack parameter.

Acceptance criteria:

1. TAs should be able to email the unique email address submitted as part of the assignment and they should be able to get reasonable predictions (spam/not spam) for the emails they send.
2. TAs should be able to stand up the CloudFormation template (T1) within a separate account, using their own prediction endpoint (E1'), and successfully test the system.
    0. This also assumes that you provide the TAs with the code for the Lambda function (LF1).

Extra credit (10 points):

Please find below the assignment prompt to receive extra credits:

In real-world applications, machine learning models are usually retrained on newly obtained data to stay updated. For extra credits, complement your spam classifier with a retraining service. To do that, user Cloudwatch and Lambda function that does the retraining and code deployment. For simplicity, retrain the model on the same data from scratch.