**MLOps Course 4 – Dog Breed**

**AWS Machine Learning Nanodegree**

**SageMaker – Notebook Setup**

I chose the instance type 'ml.m5.xlarge' after considering the cost, computing power, and speed of launching for the SageMaker notebook instance type.

**EC2**

Though the Deep Learning AMI does not support t, m or c instances. I used m5.xlarge instance for the ec2 instance since g & p instances are not enabled for students by default.

The main difference I noticed between the ec2 and sagemaker codes is the difference between a python script and a jupyter notebook. The ec2train1.py file contains first the definitions of all the functions we are going to need. Also, this file cannot contain any code we don’t want to rerun each time, as how de with notebook cells.

**Lambda**

Since Lambda is only invoked by an event, it uses the runtime.invoke\_endpoint() method after it gathers the event, endpoint and runtime values. The method then invokes the endpoint by entering the event’s data which is the url of the test image as a json file. It finally decodes the result it obtains and sends it as the body of the return statement.

**Security & Testing**

I believe AWS is a very secure workspace if and only if used correctly by taking the time to specify, manage and periodically check on the roles and permissions given to them, specially in the old or unused services. The specification of the needed permissions and policies is identifying what it is exactly that our service need to know and have access for and what should it not. Not just give it ‘Full access’ then if any security issue arises it might cause bigger security issues on the data of the whole project or even company.

**Concurrency & Auto-Scaling**

I set-up provisioned concurrency as it initializes a requested number of execution environments so that they are prepared to respond immediately to my function's invocations.

While setting up auto-scaling I made the maximum\_instance\_count = 3 to be able to handle high traffic. I also only sat the target value to 15 requests so that it doesn’t wait until much requests are made to auto-scale. Finally I made the scale in and out cool down = 30 seconds so it waits until the changes persist for 30 seconds before taking an action to increase or increase instances scaling.