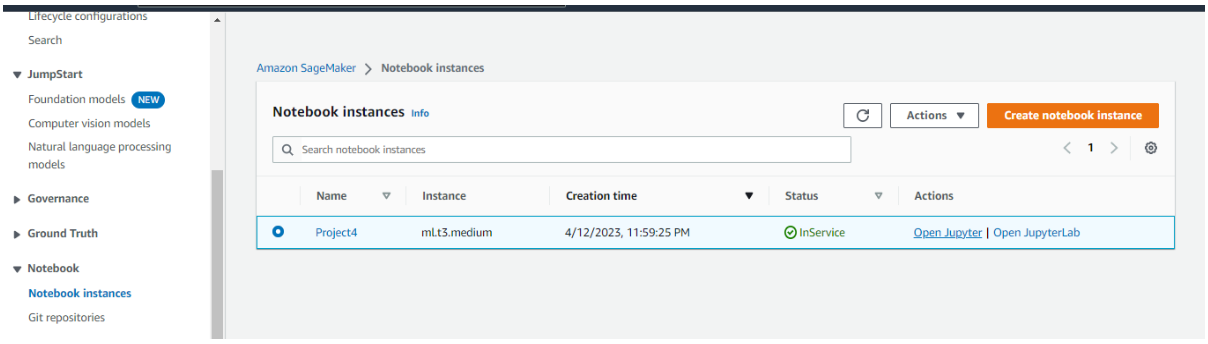
Operationalizing an AWS Machine Learning Project

1) Write a short justification for why you chose the SageMaker notebook instance type you did (consider cost computing power, and speed of launching for each instance type)

   Answer: I used the **ml.t3.medium** instance which is a general purpose instance. It provides average computing power and speed at low cost. Considering the nature of this project, there was no need for extra high performance computing, hence the choice of this instance to save cost.

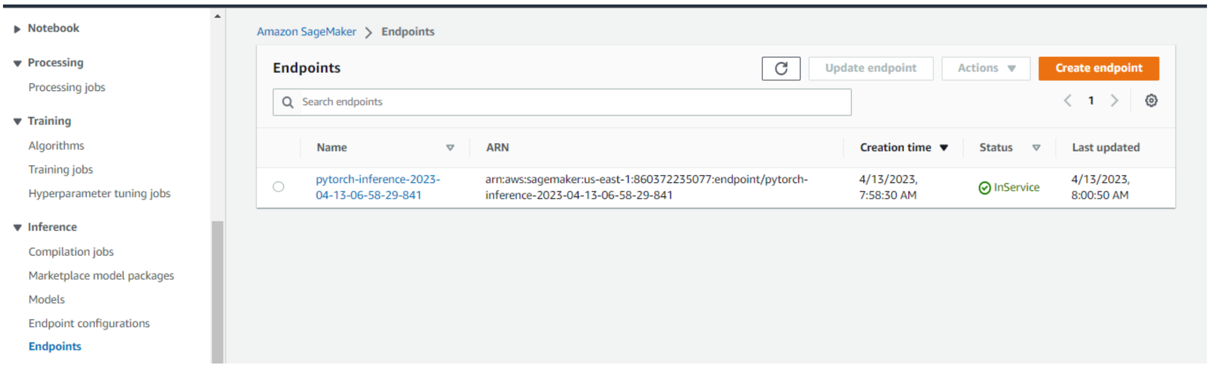
2) After launching your SageMaker instance, take a screenshot of your sagemaker dashboard's Notebooks > Instances section to show what you have done.



	Name	Instance	Creation time	Status	Actions
●	Project4	ml.t3.medium	4/12/2023, 11:59:25 PM	InService	Open Jupyter | Open JupyterLab

Fig 1: SageMaker notebook instance

3) Take a screenshot of your deployed endpoint (single instance training) and note its name

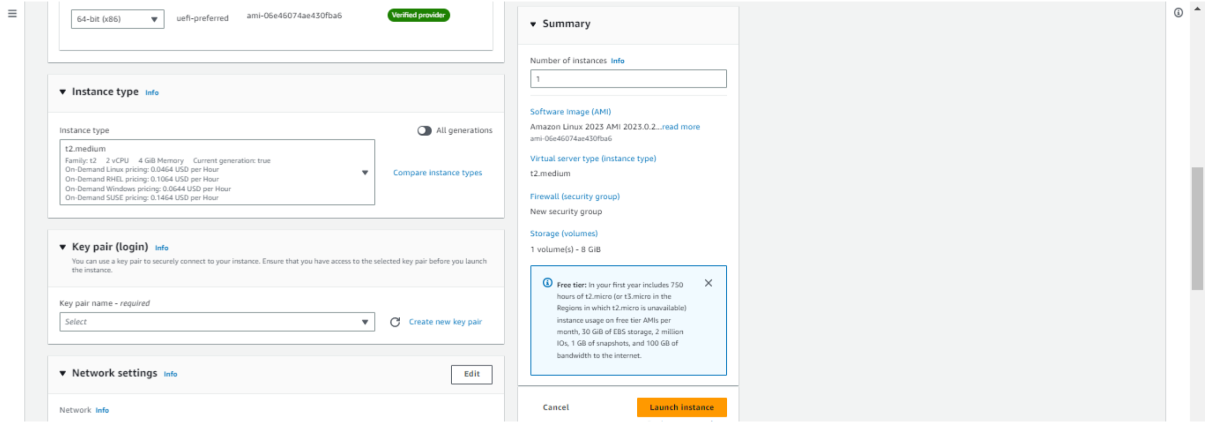


	Name	ARN	Creation time	Status	Last updated
○	pytorch-inference-2023-04-13-06-58-29-841	arn:aws:sagemaker:us-east-1:860372235077:endpoint/pytorch-inference-2023-04-13-06-58-29-841	4/13/2023, 7:58:30 AM	InService	4/13/2023, 8:00:50 AM

Fig 2: Deployed endpoint after single instance training

4) Write a justification for the type of EC2 instance you created.

Answer: Initially, I used the t2.medium instance to save cost. However, when I tried to run the solution.py file, I encountered a numpy import error and I tried to address it by activating pytorch in EC2. The EC2 t2.medium instance was unable to activate pytorch and I switched to a g3s.xlarge instance as suggested by the error message. Relevant screenshots are attached below.



64-bit (x86) ▾ uefi-preferred ami-06e46074ae430fba6 Verified provider

▾ Instance type Info

Instance type

t2.medium
Family: t2 2 vCPU 4 GiB Memory Current generation: true
On-Demand Linux pricing: 0.0464 USD per Hour
On-Demand RHEL pricing: 0.1064 USD per Hour
On-Demand Windows pricing: 0.0644 USD per Hour
On-Demand SUSE pricing: 0.1464 USD per Hour

○ All generations

Compare instance types

▾ Key pair (login) Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

Select ▾

Create new key pair

▾ Network settings Info

Edit

Network Info

▾ Summary

Number of instances Info

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.0.2...read more
ami-06e46074ae430fba6

Virtual server type (instance type)

t2.medium

Firewall (security group)

New security group

Storage (volumes)

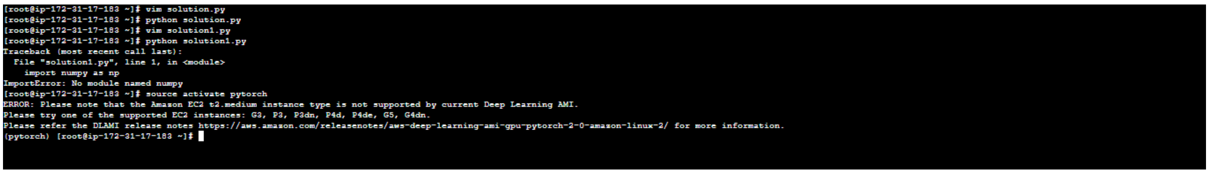
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel

Launch instance

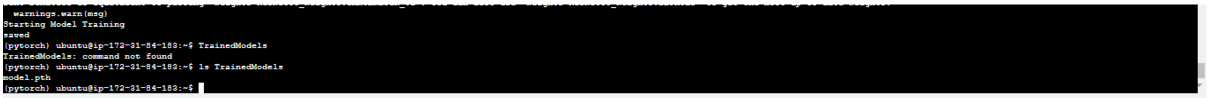
Fig 3: Initial EC2 instance



```
[root@ip-172-31-17-183 ~]# vim solution.py
[root@ip-172-31-17-183 ~]# python solution.py
[root@ip-172-31-17-183 ~]# vim solution1.py
[root@ip-172-31-17-183 ~]# python solution1.py
Traceback (most recent call last):
  File "solution1.py", line 1, in <module>
    import numpy as np
ImportError: No module named numpy
[root@ip-172-31-17-183 ~]# source activate pytorch
ERROR: Please note that the Amazon EC2 t2.medium instance type is not supported by current Deep Learning AMI.
Please try one of the supported EC2 instances: G3, P3, P3dn, P4d, P4de, G5, G4dn.
Please refer the DLAMI release notes https://aws.amazon.com/releasenotes/aws-deep-learning-ami-gpu-pytorch-2-0-amazon-linux-2/ for more information.
(pytorch) [root@ip-172-31-17-183 ~]#
```

Fig 4: EC2 error message

5) Open the TrainedModels directory in your EC2 instance and take a screenshot of the model that has been saved in it.



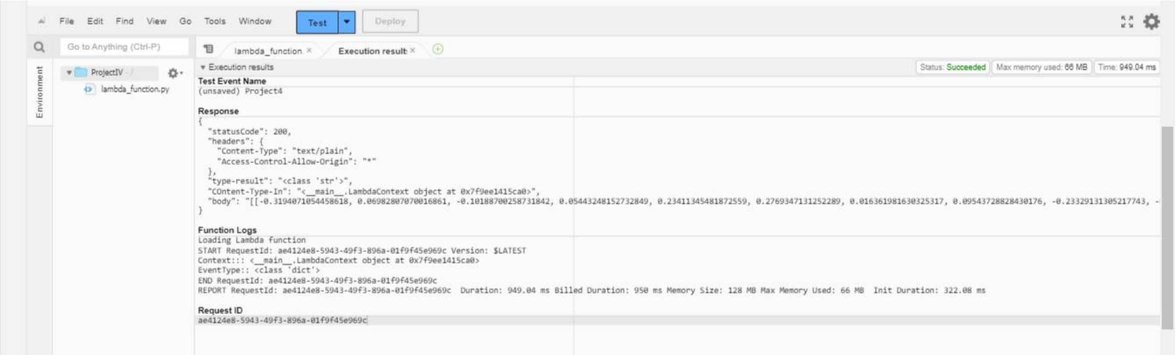
```
  warnings.warn(msg)
Starting Model Training
saved
(pytorch) ubuntu@ip-172-31-84-183:~$ TrainedModels
TrainedModels: command not found
(pytorch) ubuntu@ip-172-31-84-183:~$ ls TrainedModels
model.pth
(pytorch) ubuntu@ip-172-31-84-183:~$
```

Fig 5: Model in the TrainedModels directory

6) Write on the differences between the code in “train_and_deploy-solution.ipynb” and “ec2train1.py”

Answer: In ec2train1.py, the parameters, estimators, and tuners are not explicitly defined. ec2train1.py is optimized by building functions to handle model training and testing. These functions can be called on data contained in EC2 when needed. In the train_and_deploy-solution notebook, however, the data is imported and model training and testing is done using explicitly defined parameters, estimators, and tuners. In addition, libraries such as boto3 and sagemaker cannot be used in ec2train1.py.

7) Take a screenshot of the lambda function after running the test event, and provide the list of numbers given in the output.



Test Event Name

```
(unsaved) Project4
```

Response

```
{
  "statusCode": 200,
  "headers": {
    "Content-Type": "text/plain",
    "Access-Control-Allow-Origin": "*"
  },
  "type-result": "<class 'str'>",
  "COntent-Type-In": "<__main__.LambdaContext object at 0x7f9ee1415ca0>",
  "body": "[[-0.3194071054458618, 0.06982807070016861, -0.10188700258731842, 0.05443248152732849, 0.23411345481872559, 0.2769347131252289, 0.016361981630325317, 0.09543728828430176, -0.23329131305217743, -
}
```

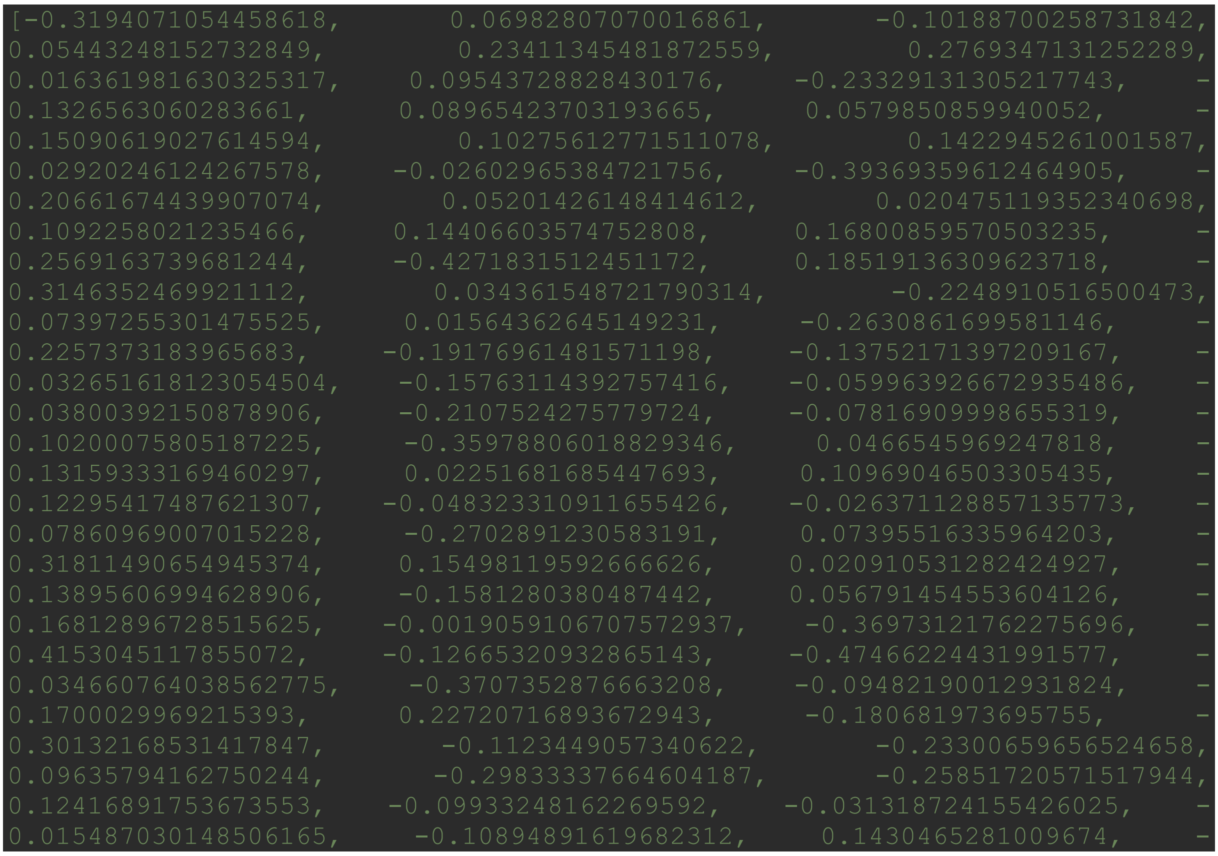
Function Logs

```
Loading Lambda function
START RequestId: ae4124e8-5943-49f3-896a-01f9f45e969c Version: $LATEST
Context::: <__main__.LambdaContext object at 0x7f9ee1415ca0>
EventType:: <class 'dict'>
END RequestId: ae4124e8-5943-49f3-896a-01f9f45e969c
REPORT RequestId: ae4124e8-5943-49f3-896a-01f9f45e969c  Duration: 949.04 ms Billed Duration: 950 ms Memory Size: 128 MB Max Memory Used: 66 MB  Init Duration: 322.08 ms
```

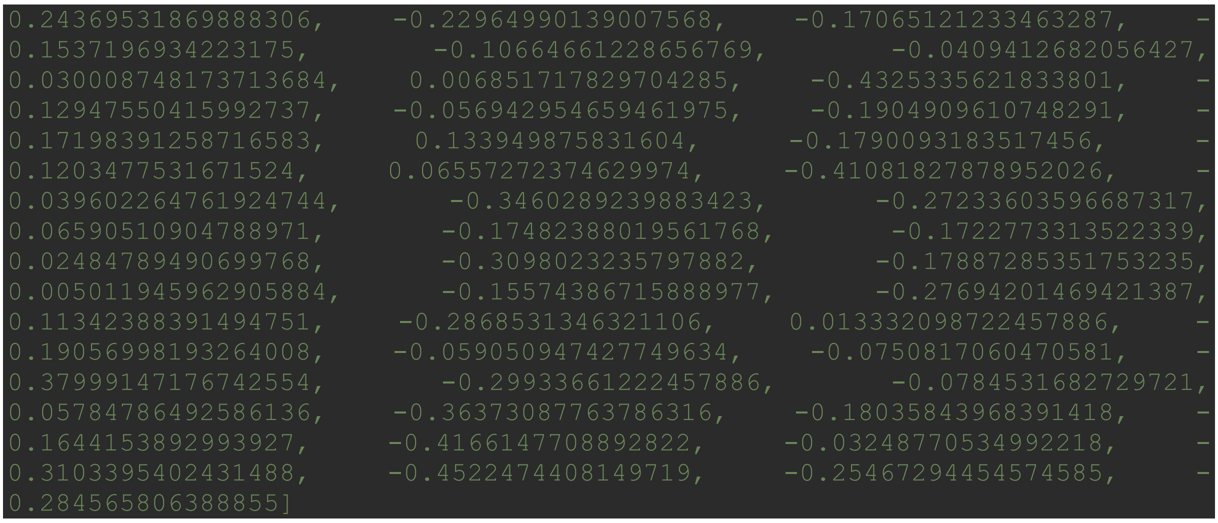
Request ID

```
ae4124e8-5943-49f3-896a-01f9f45e969c
```

Fig 6: Lambda function test output



```
[-0.3194071054458618,       0.06982807070016861,       -0.10188700258731842,
0.05443248152732849,        0.23411345481872559,        0.2769347131252289,
0.016361981630325317,    0.09543728828430176,    -0.23329131305217743,    -
0.1326563060283661,      0.08965423703193665,      0.0579850859940052,      -
0.15090619027614594,        0.10275612771511078,        0.1422945261001587,
0.02920246124267578,    -0.02602965384721756,    -0.39369359612464905,    -
0.20661674439907074,       0.05201426148414612,       0.020475119352340698,
0.1092258021235466,     0.14406603574752808,     0.16800859570503235,     -
0.2569163739681244,     -0.4271831512451172,     0.18519136309623718,     -
0.3146352469921112,       0.034361548721790314,       -0.2248910516500473,
0.07397255301475525,     0.01564362645149231,     -0.2630861699581146,     -
0.2257373183965683,    -0.19176961481571198,    -0.13752171397209167,    -
0.032651618123054504,    -0.15763114392757416,    -0.059963926672935486,    -
0.03800392150878906,    -0.2107524275779724,    -0.07816909998655319,    -
0.10200075805187225,     -0.35978806018829346,     0.0466545969247818,     -
0.13159333169460297,     0.02251681685447693,     0.10969046503305435,     -
0.12295417487621307,    -0.048323310911655426,    -0.026371128857135773,    -
0.07860969007015228,     -0.2702891230583191,     0.07395516335964203,     -
0.31811490654945374,     0.15498119592666626,     0.020910531282424927,     -
0.13895606994628906,     -0.1581280380487442,     0.056791454553604126,     -
0.16812896728515625,    -0.0019059106707572937,    -0.36973121762275696,    -
0.4153045117855072,     -0.12665320932865143,     -0.47466224431991577,     -
0.034660764038562775,    -0.3707352876663208,    -0.09482190012931824,    -
0.1700029969215393,     0.22720716893672943,     -0.180681973695755,     -
0.30132168531417847,       -0.1123449057340622,       -0.23300659656524658,
0.09635794162750244,       -0.29833337664604187,       -0.25851720571517944,
0.12416891753673553,    -0.09933248162269592,    -0.031318724155426025,    -
0.015487030148506165,     -0.10894891619682312,     0.1430465281009674,     -
```

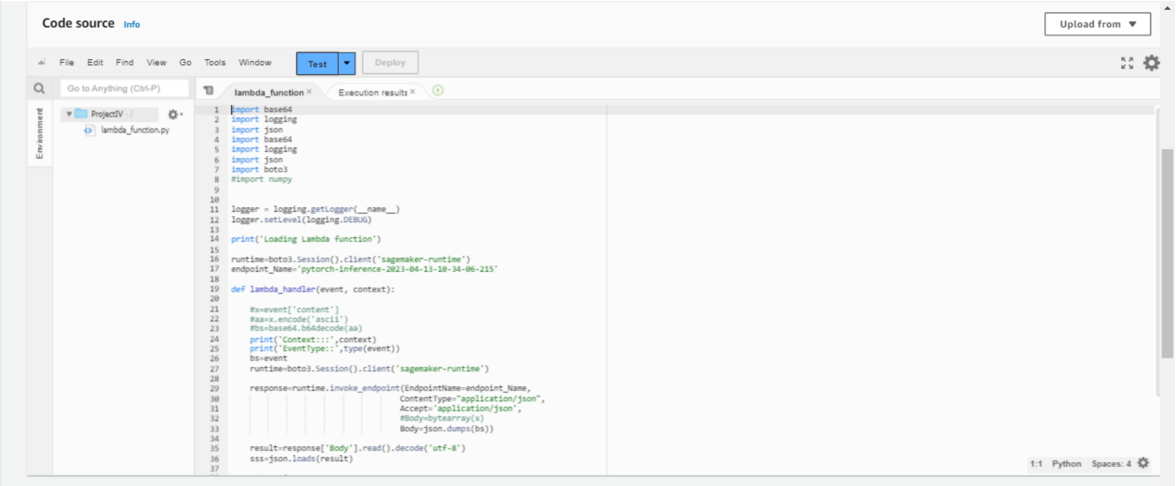


```
0.24369531869888306,    -0.22964990139007568,    -0.17065121233463287,    -
0.1537196934223175,       -0.10664661228656769,       -0.0409412682056427,
0.030008748173713684,    0.006851717829704285,    -0.4325335621833801,    -
0.12947550415992737,    -0.05694295465946197 5,    -0.19049096107 48291,    -
0.17198391258716583,     0.133949875831604,     -0.1790093183517456,     -
0.1203477531671524,     0.06557272374629974,     -0.41081827878952026,    -
0.039602264761924744,       -0.3460289239883423,       -0.27233603596687317,
0.06590510904788971,       -0.17482388019561768,       -0.172277331352 2339,
0.02484789490699768,       -0.3098023235797882,       -0.17887285351753235,
0.005011945962905884,       -0.15574386715888977,       -0.27694201469421387,
0.11342388391494751,     -0.2868531346321106,     0.013332098722457886,     -
0.19056998193264008,    -0.05905094742774963 4,    -0.0750817060470581,    -
0.37999147176742554,       -0.29933661222457886,       -0.0784531682729721,
0.05784786492586136,    -0.36373087763786316,    -0.18035843968391418,    -
0.1644153892993927,    -0.4166147708892822,    -0.03248770534992218,    -
0.3103395402431488,    -0.4522474408149719,    -0.25467294454574585,    -
0.284565806388855]
```

Fig 7: List of numbers in Lambda function test output

8) Describe how the Lambda function is written and how it works

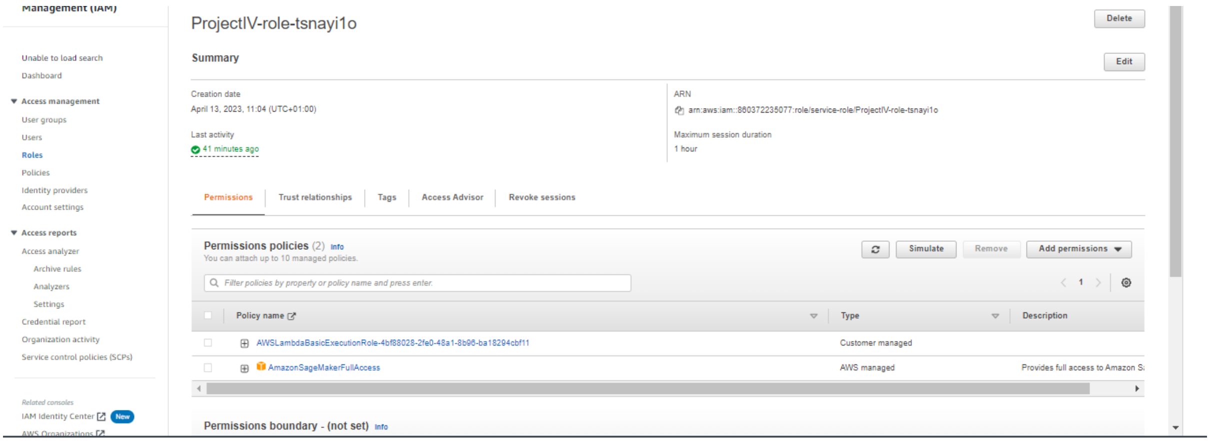
Answer: The lamda_handler function is the major component of this Lambda function. It takes in a picture (passed in as a url using the JSON file of the Lambda function) which it passes through the model contained in a SageMaker endpoint (the endpoint is defined within the Lambda function as endpoint_Name). The model contained in this specified endpoint is a dog breed classifier. Therefore, the output of this Lambda function is a list containing the probability of the dog in the input picture belonging to the various breeds in the dataset.



```
import base64
import logging
import json
import base64
import logging
import json
import boto3
#import numpy


logger = logging.getLogger(__name__)
logger.setLevel(logging.DEBUG)

print('Loading Lambda function')

runtime=boto3.Session().client('sagemaker-runtime')
endpoint_Name='pytorch-inference-2023-04-13-10-34-06-215'

def lambda_handler(event, context):

    #x=event['content']
    #aa=x.encode('ascii')
    #bs=base64.b64decode(aa)
    print('Context:::',context)
    print('EventType::',type(event))
    bs=event
    runtime=boto3.Session().client('sagemaker-runtime')

    response=runtime.invoke_endpoint(EndpointName=endpoint_Name,
                                    ContentType="application/json",
                                    Accept='application/json',
                                    #Body=bytearray(x)
                                    Body=json.dumps(bs))

    result=response['Body'].read().decode('utf-8')
    sss=json.loads(result)

```

Fig 8: Lambda function

9) Take a screenshot of the policies attached to your Lambda function role



ProjectIV-role-tsnayi1o

Summary

Creation date
April 13, 2023, 11:04 (UTC+01:00)

Last activity
41 minutes ago

ARN
arn:aws:iam::860372235077:role/service-role/ProjectIV-role-tsnayi1o

Maximum session duration
1 hour

Permissions policies (2)
You can attach up to 10 managed policies.

Filter policies by property or policy name and press enter.

Policy name	Type	Description
AWSLambdaBasicExecutionRole-4bf88028-2fe0-48a1-8b96-ba18294cbf11	Customer managed	
AmazonSageMakerFullAccess	AWS managed	Provides full access to Amazon S

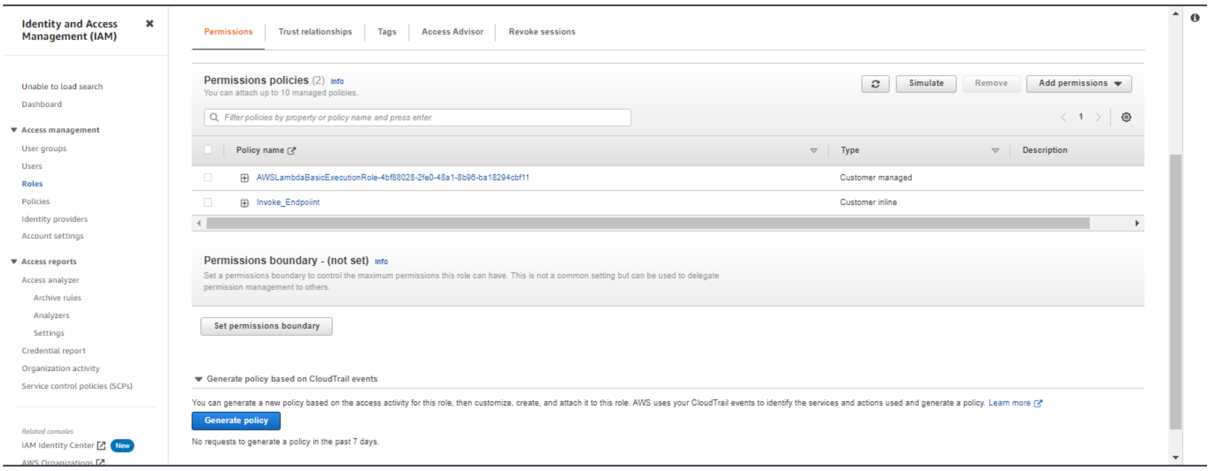
Permissions boundary - (not set)

Fig 9: Policies attached to the Lambda function role before security considerations

10) Write about the security vulnerabilities in your AWS workspace

Answer: Attaching the AmazonSageMakerFullAccess policy to my Lambda function exposes my account to security risks through the function. This is because the Lambda function has access to all my endpoints in SageMaker.

To address this issue, I replaced the AmazonSageMakerFullAccess policy with a policy that allows my Lambda function access to only the pytorch inference endpoint that it passes data to. Appropriate screenshots are attached below.



Permissions policies (2)
You can attach up to 10 managed policies.

Filter policies by property or policy name and press enter.

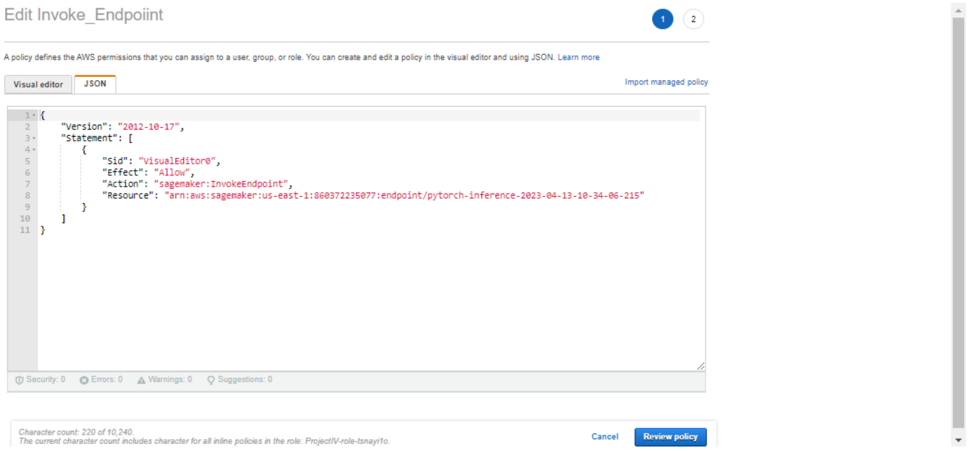
Policy name	Type	Description
AWSLambdaBasicExecutionRole-4bf88028-2fe0-48a1-8b96-ba18294cbf11	Customer managed	
Invoke_Endpoiint	Customer inline	

Permissions boundary - (not set)
Set a permissions boundary to control the maximum permissions this role can have. This is not a common setting but can be used to delegate permission management to others.

Generate policy based on CloudTrail events
You can generate a new policy based on the access activity for this role, then customize, create, and attach it to this role. AWS uses your CloudTrail events to identify the services and actions used and generate a policy.

No requests to generate a policy in the past 7 days.

Fig 10: Policies attached to the Lambda function role after security considerations



Edit Invoke_Endpoiint

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. Learn more

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "sagemaker:InvokeEndpoint",
            "Resource": "arn:aws:sagemaker:us-east-1:860372235077:endpoint/pytorch-inference-2023-04-13-10-34-06-215"
        }
    ]
}
```

Character count: 220 of 10,240.
The current character count includes character for all inline policies in the role: ProjectIV-role-tsnayi1o.

Fig 11: JSON for new policy attached to Lambda function

11) Write about the choices you made during concurrency and autoscaling setup

Answer:

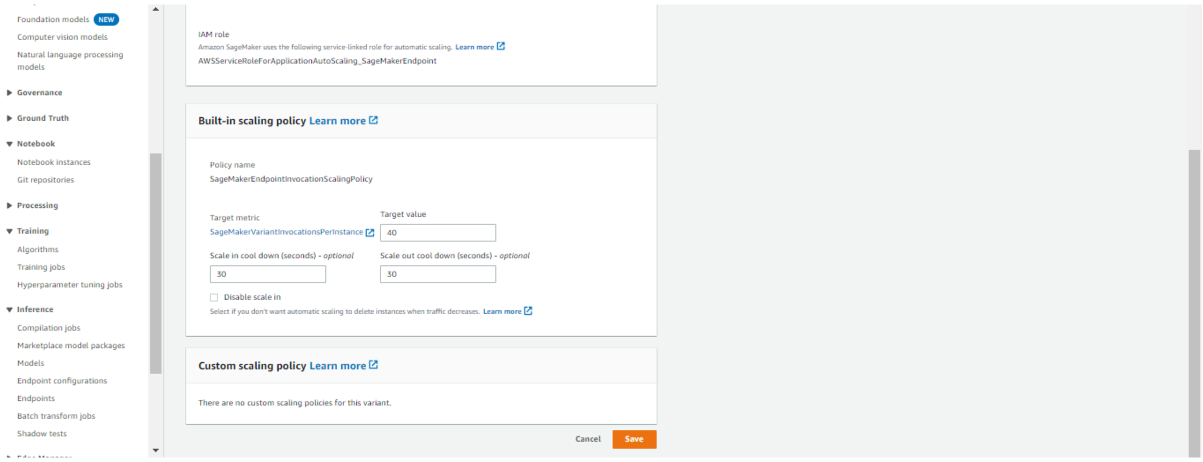
Autoscaling: I set the target value to 40. This means that instances will be added when there are greater than or equal to 40 simultaneous requests to this endpoint. I set the value high to ensure that new instances are not activated unless absolutely necessary, ultimately leading to less cost.

I set the maximum instance count to 2 because I do not expect high traffic on my endpoint.

I set 30 for the scale in cool down value to ensure that extra resources are not employed for momentary spikes in request traffic. 30 minutes is enough time to ascertain that there is indeed high traffic, before deploying more resources.

I also set 30 for the scale out cool down value to ensure that resources are not retrieved for temporary reductions in request traffic. 30 minutes is sufficient time to ensure that there is indeed traffic reduction before retrieving deployed resources.

Concurrency: Since I do not expect high traffic on this project and provisioned concurrency needs to be lower than reserved concurrency, I picked 5 reserved instances and 3 provisioned instances to save cost.



IAM role

Amazon SageMaker uses the following service-linked role for automatic scaling.

AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint

Built-in scaling policy

Policy name

SageMakerEndpointInvocationScalingPolicy

Target metric: SageMakerVariantInvocationsPerInstance

Target value: 40

Scale in cool down (seconds) - *optional*: 30

Scale out cool down (seconds) - *optional*: 30

□ Disable scale in

Select if you don't want automatic scaling to delete instances when traffic decreases.

Custom scaling policy

There are no custom scaling policies for this variant.

Fig 12: Autoscaling configuration