1. Choosing the Instance type for the Sagemaker instance
    - Choosing “ml.t3.xlarge” as the instance type was because it was relatively cheap and had enough memory size to speed up the running of codes in the notebook instance.

2. Choosing the instance type for the EC2 instanc3
    - The choice of instance type for the EC2 instance was because it was reasonably cheap, costing only 0.1208 USD per hour, and with enough memory to speed up the work.

3. Difference between code in ec2train1.py and hpo.py
    - In the hpo.py file for step 1, the arguments in the functions were accessed and updated using parser arguments; in the ec2train.py function however, these arguments were defined in the same file.

4. How the lambda is written and how it works
    - The lambda function is written to receive an event or request in the form of a URL. It then invokes an endpoint and passes the event to the endpoint to receive a response. The response representing the prediction of the event or request is then returned in a json format together with the data type of the response.

5. Lambda function returns:
    - "body": "[[-7.323208808898926, -5.868058681488037, -3.6986989974975586, -2.3293845653533936, -4.757060527801514, -7.977996826171875, -2.8250021934509277, -3.1752026081085205, -9.41547679901123, -0.5277163982391357, -1.834295392036438, -4.956415176391602, -1.814689040184021, 0.038602158427238464, -5.542778968811035, -3.8389194011688232, -7.565074920654297, -6.438897609710693, -5.6794819831848145, 0.9319513440132141, -6.448575496673584, -2.147934913635254, -6.969007968902588, -11.310032844543457, -7.261974334716797, -9.048237800598145, -2.7895143032073975, -3.6819655895233154, -6.31048059463501, -2.9175972938537598, -4.614521503448486, -3.436250686645508, -7.632781505584717, -3.3621976375579834, -10.105517387390137, -4.259114742279053, -4.6781005859375, -3.9099559783935547, -2.433901309967041, -3.3495137691497803, -3.7128231525421143, -3.6016976833343506, 0.2475094050168991, -5.5424017906188965, -2.0727641582489014, -8.386947631835938, -4.814947605133057, -4.09656286239624, -5.506553649902344, -4.020261287689209, -4.311730861663818, -6.611286640167236, -7.0482177734375, -3.1082332134246826, -

9.441901206970215, -3.700179100036621, -7.471911907196045, -6.17254638671875, -4.533364295959473, -3.886809825897217, -7.456058979034424, -6.630995750427246, -8.573760032653809, -9.163514137268066, -5.377211570739746, -8.366514205932617, -1.4861317873001099, -5.4418511390686035, -2.5826752185821533, -2.638739585876465, -0.29559311270713806, -6.345597267150879, -7.187933921813965, -7.579967975616455, -6.980849742889404, -4.5978827476501465, -7.751783847808838, -4.970809459686279, -5.927196979522705, -6.545125961303711, -0.6157044768333435, -8.8350248336792, 0.5154339075088501, -1.2690186500549316, -7.910295486450195, -6.039488315582275, -2.981978416442871, -7.055603981018066, -3.3975753784179688, -4.20492696762085, -6.902423858642578, -5.580059051513672, -6.6517181396484375, -8.013251304626465, -5.8466267585754395, -3.311131000518799, -3.140770673751831, -4.6263322830200195, -7.29825496673584, -7.395380020141602, -12.725812911987305, -4.2700371742248535, -4.035181999206543, -4.833872318267822, -7.1838531494140625, -7.844658851623535, -5.026651382446289, -1.8399213552474976, -3.1045444011688232, -2.0579233169555664, -3.344735860824585, -2.680513620376587, -8.207093238830566, -4.858431816101074, -8.078134536743164, -3.044567108154297, -8.184635162353516, -2.7655506134033203, -4.891141414642334, -0.3080110251903534, -3.205339193344116, -4.836352348327637, -5.215157985687256, -5.716874122619629, -8.553523063659668, -4.657400608062744, -4.009560585021973, -1.8173075914382935, -4.761080265045166, -6.040896415710449, -6.786423206329346, -4.0866923332214355, -4.755620956420898]]"

6. AWS workspace security and vulnerabilities
    - The was no security issue in my opinion since lambda function role was only granted full access to the SageMaker, to enable it to invoke the endpoint.

7. Setting up Concurrency and Auto-Scaling
    - I set up a reserved concurrency of five instances to control high throughput. I chose reserved over provisional concurrency since it is the cheaper option.
    - I set up auto-scaling with a maximum of five instance count, a scale-in cool down time of 30 seconds and a scale-out cool down time of thirty seconds. This will enable a quick response to a case of high traffic while making sure that when the traffic goes down, the extra instances are immediately turned off. I also specified a target value of 30 to allow the extra instances to be turned on only when 30 or more requests are received simultaneously. This is to avoid unnecessary creation of extra instance to add up unnecessary cost.