Step 1:

1. I choose ml.t3.medium instance type for my Jyputer because first it’s cheap only coat $0.05 per hour ,second It supports fast lunch, third its computing power is good enough for this notebook (2 vCPU +4 GIB) and I used it before in my previous project and worked well with me.

1. For the multi-instance-training I used instance count = 2 and the instance type (ml.g4dn.xlarge).
2. The first Endpoint (pytorch-inference-2022-10-19-06-56-41-516)
3. The multi-training Endpoint (pytorch-inference-2022-10-19-07-56-20-741)

Step 2:

1. I selected the t3.medium instance type for my EC2 instance because its low cost .0416 per hour ,computing power 2vCPU + 4GIB. Compared with the cost and the computing power I think it’s a good option to choose and it’s enough for the work required. It worked really well on the script.
2. The differences between train\_and\_deploy and ec2train1.py, the first one is designed code for sagemaker, the second one is for EC2 , that’s why there are no estimator, as example to chose instance , instance type and hyperparameters . Instead, we took the training script (the entry point for sagemaker code, hpo) and now we apply it directly on the EC2 instance after adjusting. Another thing the deploy method is different between sagemaker and EC2 because in EC2 It's not possible to deploy directly.

Step 3:

1. This lambda function is used to invoke the endpoint that I created before and pass input to it, and see if it’s returning output. first we imported the packages we needs , then we made variable takes the endpoint that needs to invoke , then we made the lambda handler function. We used the invoke\_endpoint()method and pass to it the endpoint name, content type as json format and the body variable which will use the event (input)that will be tested, then we made variable results that contains of .read() method and the decode() method for the input we want that will be read and tested by the invoking of our endpoint. The function will return to us type of results, context and the body which is the result of the machine learning output from my deployed endpoint as result.

Step 4:

1. The lambda test event output: [[0.32415467500686646, -0.03945959359407425, -0.32424771785736084, 0.20683303475379944, 0.4182508885860443, 0.14003846049308777, -0.22266043722629547, 0.2730558216571808, -0.5274960994720459, 0.1156652420759201, 0.08736374974250793, 0.10468190163373947, 0.01654638722538948, 0.2893403470516205, 0.09673149883747101, 0.052534181624650955, 0.46830737590789795, 0.501164972782135, -0.33419355750083923, -0.037554606795310974, 0.011357570067048073, -0.4087024927139282, 0.47408461570739746, 0.07646151632070541, -0.2926192879676819, -0.5465134382247925, -0.0036054030060768127, -0.4604722857475281, 0.3557682931423187, -0.027024609968066216, 0.00022350717335939407, 0.019998304545879364, -0.4213276505470276, -0.10173448175191879, 0.18596619367599487, 0.31655827164649963, 0.2415444552898407, 0.2623886168003082, 0.3346058130264282, -0.2166527509689331, 0.09419950097799301, 0.19104889035224915, -0.019041871652007103, 0.028937505558133125, -0.13864967226982117, 0.614077091217041, -0.3700203001499176, 0.11308257281780243, 0.16403762996196747, -0.24541477859020233, 0.15524160861968994, 0.18015968799591064, 0.10495049506425858, 0.37399324774742126, -0.002842813730239868, 0.30662664771080017, 0.3267952501773834, 0.19176587462425232, -0.3872632384300232, 0.40673357248306274, 0.342685729265213, 0.21957676112651825, 0.42723318934440613, -0.21292629837989807, -0.13774016499519348, -0.3635064363479614, -0.6460488438606262, 0.1410589963197708, 0.04328291863203049, -0.2592546045780182, 0.035035423934459686, -0.29934245347976685, -0.2960464358329773, -0.2270507514476776, -0.08166112005710602, 0.3286653161048889, -0.34085339307785034, -0.042911022901535034, 0.3474991023540497, -0.07673777639865875, -0.02821299433708191, 0.18192629516124725, -0.30132409930229187, -0.2850528061389923, -0.22442318499088287, -0.04993318393826485, 0.014569427818059921, -0.07600665837526321, 0.28466498851776123, 0.34663325548171997, -0.2407742738723755, 0.27541500329971313, -0.7116633057594299, 0.01276586577296257, 0.3407488167285919, -0.12125016003847122, -0.3154199421405792, 0.08883443474769592, -0.16940270364284515, -0.7831739187240601, 0.02547561749815941, -0.11944971978664398, 0.11477090418338776, -0.5440657734870911, -0.21917131543159485, 0.06461059302091599, 0.0846530869603157, -0.8033980131149292, 0.3487017750740051, -0.4073294699192047, 0.11235329508781433, -0.08344517648220062, -0.30535170435905457, -0.2744613289833069, 0.1749071180820465, -0.5934415459632874, 0.21975602209568024, 0.19156034290790558, -0.5291808843612671, 0.02977820485830307, -0.8100306987762451, -0.4970850646495819, -0.45385152101516724, 0.0844871997833252, -0.60318523645401, -0.29956328868865967, 0.05834440886974335, -0.7381641268730164, -0.2643773853778839, -0.06600257754325867, -0.43585333228111267, -0.30853116512298584, -0.7726637125015259]]
2. The IAM roles for my lambda, I gave it sagemaker full access, I think it will be better practice to just attach polices that will give access to the lambda function to invoke the endpoint only to ensure that it’s not give full permission to sagemaker that will increase the security for my account, and this for all roles gives attach only policies that required for specific access. Second my IAM dash board there are many roles that made before and no longer need to keep, that for sure is red flag for my account security and it makes security levels low and my account is in danger. For example (because they are many) , sagmaker role that I used before 2 months ExecutionRole-20220819T133486, it means in real life that it was for one of my colleagues that is not working on it anymore , it must be deleted . the are also old lambda function data-generation-role-pj3v0z1t, it must be deleted because anyway there are no such lambda function right now except for this project lambda only. Vocstartsoft, voclabs , vocareum they are roles need access to delete , I don’t know what are this roles but I think because it’s a student account and they are belong to Udacity. In less words, delete not used roles and keeps only the active ones, check the logs and see if there any weird actions in it that I didn’t do before, limit the attached policies to give access for the job that needed only will makes the security level much better to keep my account and its content, data, models to be safe.

Step 5:

1. For concurrency I choose Provisioned concurrency because it doesn’t require wait for startup time when the number of requests increasing it costs more than Reserved concurrency but its better from my opinion to avoid latency for my project because the instances always on and will be ready to respond to very have traffic scenarios. I made version and sit it up to use 5 concurrency account charges will be $6.98 per month in addition for duration and requests for my lambda, so I avoid latency when it comes to increasing traffic. However, it can be changed in the future in more or less traffic that I expect for my project.
2. For auto-scaling I choose the number of instances between 1 to 5 as max instance which means that when the traffic will increase that my endpoint can scale to 5 instances which will lead to low latency, the target value for 20 that means every 20 request could be requested at the same time for my endpoint will start to use another instance , the scale-in cooldown to 30 sec so that when increased traffic it will just take 30 sec to deploy new instance and the scale-out to 120 sec so when the traffic become lower it will take 2 min to delete instance.

These choices I made to imagine that in scenario that my project can have an ahigh volume of traffic (the project will be ready for high, unexpected traffic), so to avoid low latency and bottlenecks with my project. Of course, that will depend on the budget for my project that calculated plus the expected number of traffic could will be.

Also, in my opinion it will be much better to adjust the project so it can process high volume of traffic, even if it will be higher cost to avoid latency because if the project isn’t ready for high, unexpected traffic. The users will not be interested in it and will use another.