DSC 102: Systems for Scalable Analytics

Programming Assignment 0

1 Introduction

The goal of this programming assignment is to get you comfortable with datasets that do not fit in single-node memory and are too big for tools like Pandas or NumPy. You will be using Dask library to explore secondary storage aware data access on a single machine. In this assignment, you will be learning to setup dask on AWS and computing several descriptive statistics about the data to build intuitions for feature engineering for the final assignment.

2 Dataset Description

You are provided with the Amazon Reviews dataset with the *reviews* table as CSV file. The schemas are provided in Table 1. The dataset is available on the s3 bucket: `s3://dsc102-public`.

Column name	Column description	Example
reviewerID	ID of the reviewer	A32DT10X9WS4D0
asin	ID of the product	B003VX9DJM
reviewerName	name of the reviewer	Slade
helpful	helpfulness rating of the review	[0, 0]
reviewText	text of the review	this was a gift for my friend who loves touch lamps.
overall	rating of the product	1
summary	summary of the review	broken piece
unixReviewTime	summary of the review	1397174400
reviewTime	time of the review (raw)	04 11, 2014

Table 1: Schema of Reviews table

3 Tasks

You will use the *reviews* table to explore features related to users. Specifically, you will create the users table with the schema given in Table 2.

A code stub with function signature for this task has been provided to you. The input to the function is the reviews CSV file and you will be carrying out a series of transformations to produce the users table as DataFrame. Plug in the DataFrame you obtained as a result in <`YOUR_USERS_DATAFRAME`> and write this to `results_PA0.json` file. We will time the execution of the function `PA0`.

We have shared with you the "development" dataset and our accuracy results. Our code's runtime on 1 node is roughly 615s. You can use this to validate your results and debug your code. The final evaluation will happen on separate held-out test sets. The runtime will be different for the held-out test set.

4 Deliverables

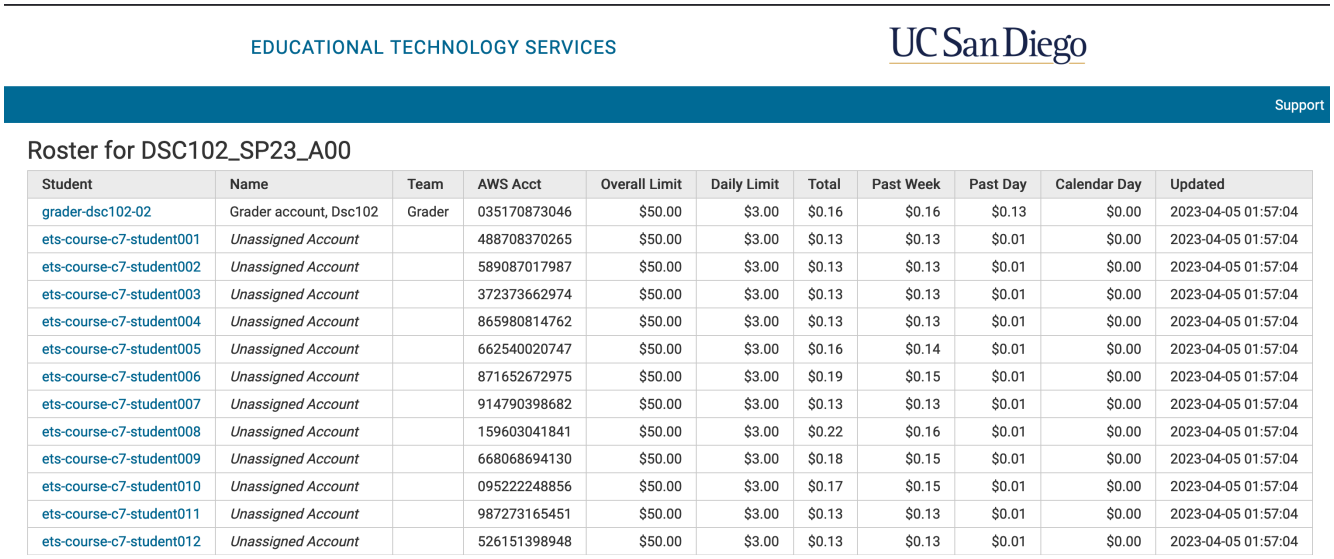
Submit your source code as <`YOUR-TEAM-ID`>`.py` on Canvas. Your source code must confirm to the function signatures provided to you. Make sure that your code is writing results to `results_PA0.json`.

Column name	Column description
reviewerID (PRIMARY KEY)	ID of the reviewer
number_products_rated	Total number of products rated by the reviewer
avg_ratings	Average rating given by the reviewer across all the reviewed products
reviewing_since	The year in which the user gave their first review
helpful_votes	Total number of helpful votes received for the users' reviews
total_votes	Total number of votes received for the users' reviews

Table 2: Schema of users table

5 Getting Started

1) Once we have set up your groups in canvas, a role will be created in AWS associated with your group. Access your AWS account using single sign-on ID: `https://ets-apps.ucsd.edu/individual/DSC102_SP23_A00/`.



EDUCATIONAL TECHNOLOGY SERVICES

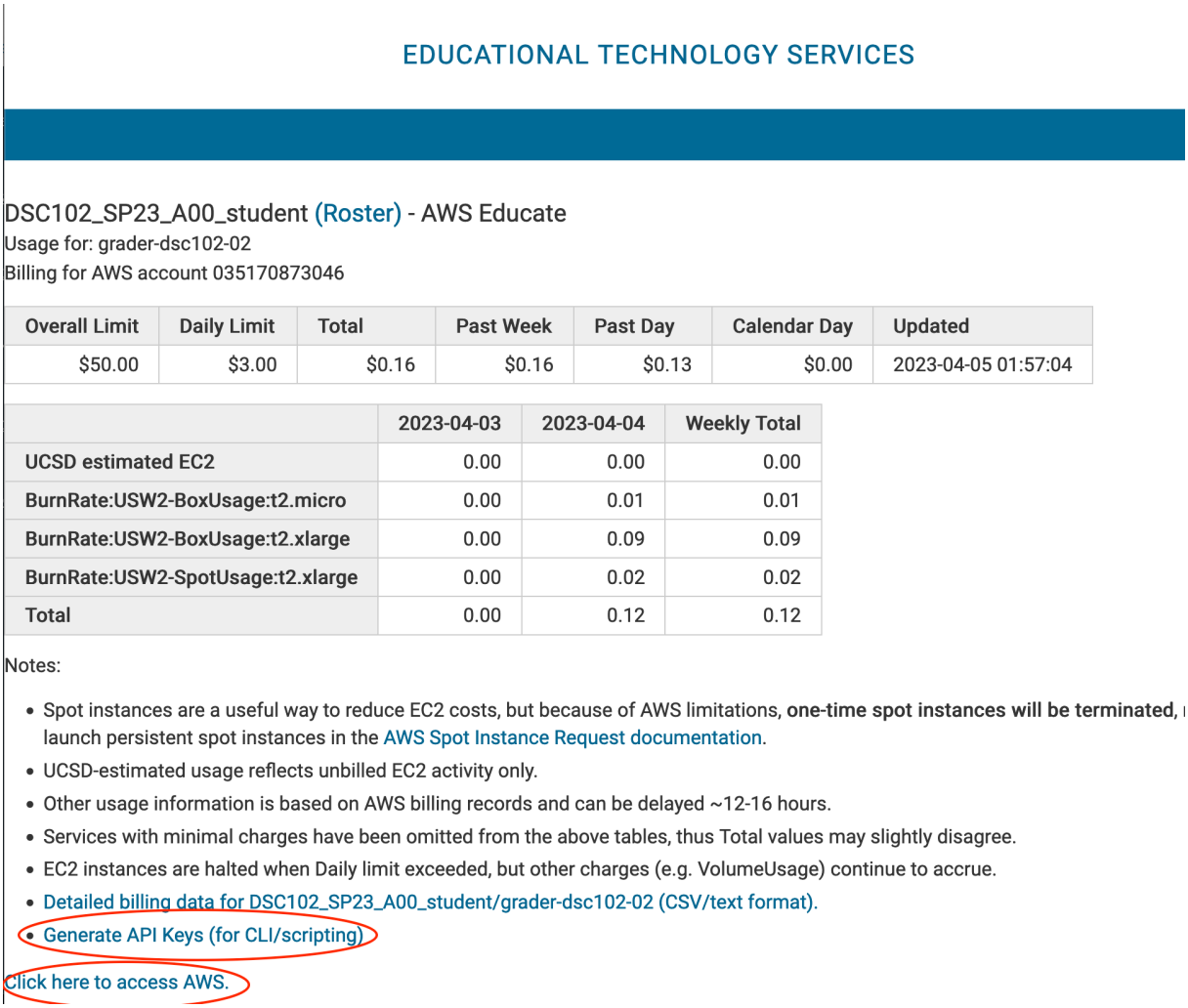
UC San Diego

Support

Roster for DSC102_SP23_A00

Student	Name	Team	AWS Acct	Overall Limit	Daily Limit	Total	Past Week	Past Day	Calendar Day	Updated
grader-dsc102-02	Grader account, Dsc102	Grader	035170873046	$50.00	$3.00	$0.16	$0.16	$0.13	$0.00	2023-04-05 01:57:04
ets-course-c7-student001	*Unassigned Account*		488708370265	$50.00	$3.00	$0.13	$0.13	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student002	*Unassigned Account*		589087017987	$50.00	$3.00	$0.13	$0.13	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student003	*Unassigned Account*		372373662974	$50.00	$3.00	$0.13	$0.13	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student004	*Unassigned Account*		865980814762	$50.00	$3.00	$0.13	$0.13	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student005	*Unassigned Account*		662540020747	$50.00	$3.00	$0.16	$0.14	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student006	*Unassigned Account*		871652672975	$50.00	$3.00	$0.19	$0.15	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student007	*Unassigned Account*		914790398682	$50.00	$3.00	$0.13	$0.13	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student008	*Unassigned Account*		159603041841	$50.00	$3.00	$0.22	$0.16	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student009	*Unassigned Account*		668068694130	$50.00	$3.00	$0.18	$0.15	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student010	*Unassigned Account*		095222248856	$50.00	$3.00	$0.17	$0.15	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student011	*Unassigned Account*		987273165451	$50.00	$3.00	$0.13	$0.13	$0.01	$0.00	2023-04-05 01:57:04
ets-course-c7-student012	*Unassigned Account*		526151398948	$50.00	$3.00	$0.13	$0.13	$0.01	$0.00	2023-04-05 01:57:04

Select your group name from the menu and you will find a summary page indicating your overall budget, daily budget, and usage. You will also find a breakdown of costs. Click the 'Click here to access AWS' link at the very bottom of the page to access the AWS console, or alternatively click the 'Generate API KEYS (for CLI/scripting)' to get credentials for the AWS command line interface. More information on the AWS command line interface can be found here: `https://aws.amazon.com/cli/`



EDUCATIONAL TECHNOLOGY SERVICES

DSC102_SP23_A00_student (Roster) - AWS Educate

Usage for: grader-dsc102-02

Billing for AWS account 035170873046

Overall Limit	Daily Limit	Total	Past Week	Past Day	Calendar Day	Updated
$50.00	$3.00	$0.16	$0.16	$0.13	$0.00	2023-04-05 01:57:04

	2023-04-03	2023-04-04	Weekly Total
UCSD estimated EC2	0.00	0.00	0.00
BurnRate:USW2-BoxUsage:t2.micro	0.00	0.01	0.01
BurnRate:USW2-BoxUsage:t2.xlarge	0.00	0.09	0.09
BurnRate:USW2-SpotUsage:t2.xlarge	0.00	0.02	0.02
Total	0.00	0.12	0.12

Notes:

- Spot instances are a useful way to reduce EC2 costs, but because of AWS limitations, **one-time spot instances will be terminated,** launch persistent spot instances in the AWS Spot Instance Request documentation.
- UCSD-estimated usage reflects unbilled EC2 activity only.
- Other usage information is based on AWS billing records and can be delayed ~12-16 hours.
- Services with minimal charges have been omitted from the above tables, thus Total values may slightly disagree.
- EC2 instances are halted when Daily limit exceeded, but other charges (e.g. VolumeUsage) continue to accrue.
- Detailed billing data for DSC102_SP23_A00_student/grader-dsc102-02 (CSV/text format).
- Generate API Keys (for CLI/scripting)

Click here to access AWS.

2) We have setup the Dask environment on an AMI with name "dsc102-dask-environment-public." Go to "AMIs" (under "Images") in your EC2 dashboard, select public images, and then search by name to find it. Select this AMI and click 'Launch Instance from AMI'. See Figure 1 and Figure 2 .

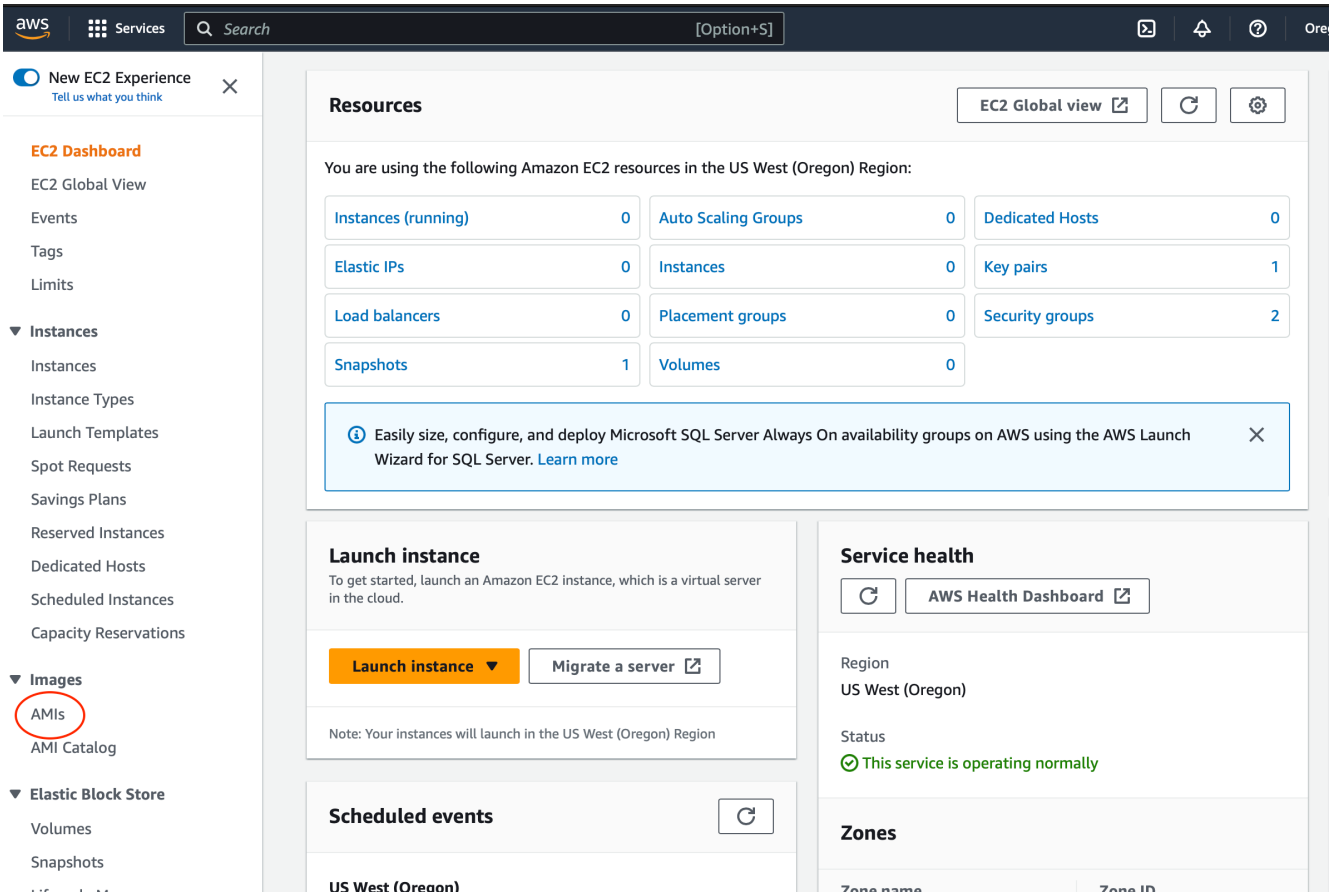

aws
Services
Search
[Option+S]
New EC2 Experience
Tell us what you think
EC2 Dashboard
EC2 Global View
Events
Tags
Limits
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Scheduled Instances
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Resources
EC2 Global view
You are using the following Amazon EC2 resources in the US West (Oregon) Region:
Instances (running) 0
Auto Scaling Groups 0
Dedicated Hosts 0
Elastic IPs 0
Instances 0
Key pairs 1
Load balancers 0
Placement groups 0
Security groups 2
Snapshots 1
Volumes 0
Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more
Launch instance
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.
Launch instance
Migrate a server
Note: Your instances will launch in the US West (Oregon) Region
Service health
AWS Health Dashboard
Region
US West (Oregon)
Status
This service is operating normally
Scheduled events
Zones
US West (Oregon)

Figure 1

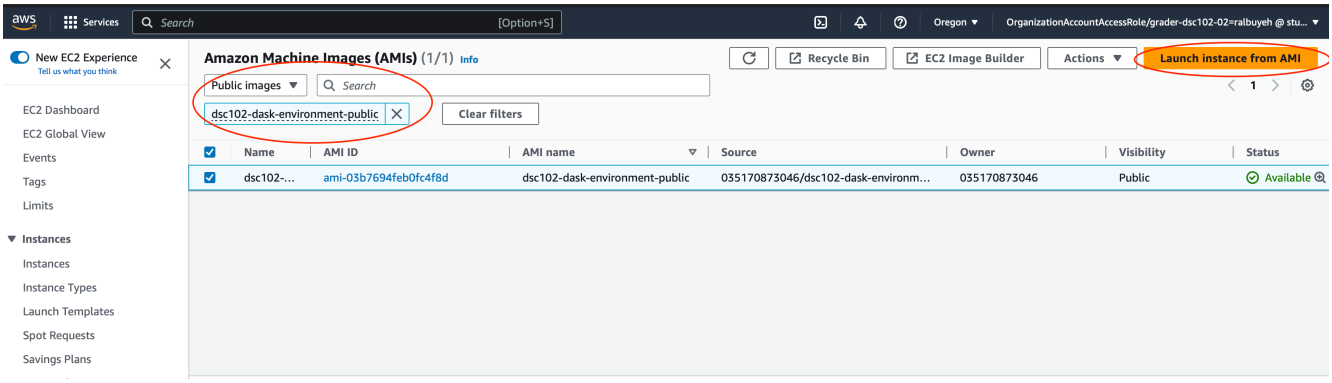


Figure 2

3) You will be launching one EC2 *Spot* instance that will be used to run dask remotely (in the cloud, not on your personal machine). Note that an AWS spot instance is heavily discounted in price, in exchange for giving AWS permissions to shut down your instance if demand for compute is high. Be mindful about backing up your code and associated artifacts.

a) You should now be on the 'Launch an Instance' page, as indicated in Figure 3. Under 'Name', give your instance a name you will remember. Under 'Number of instances' on the right side of the page, leave the value as 1.

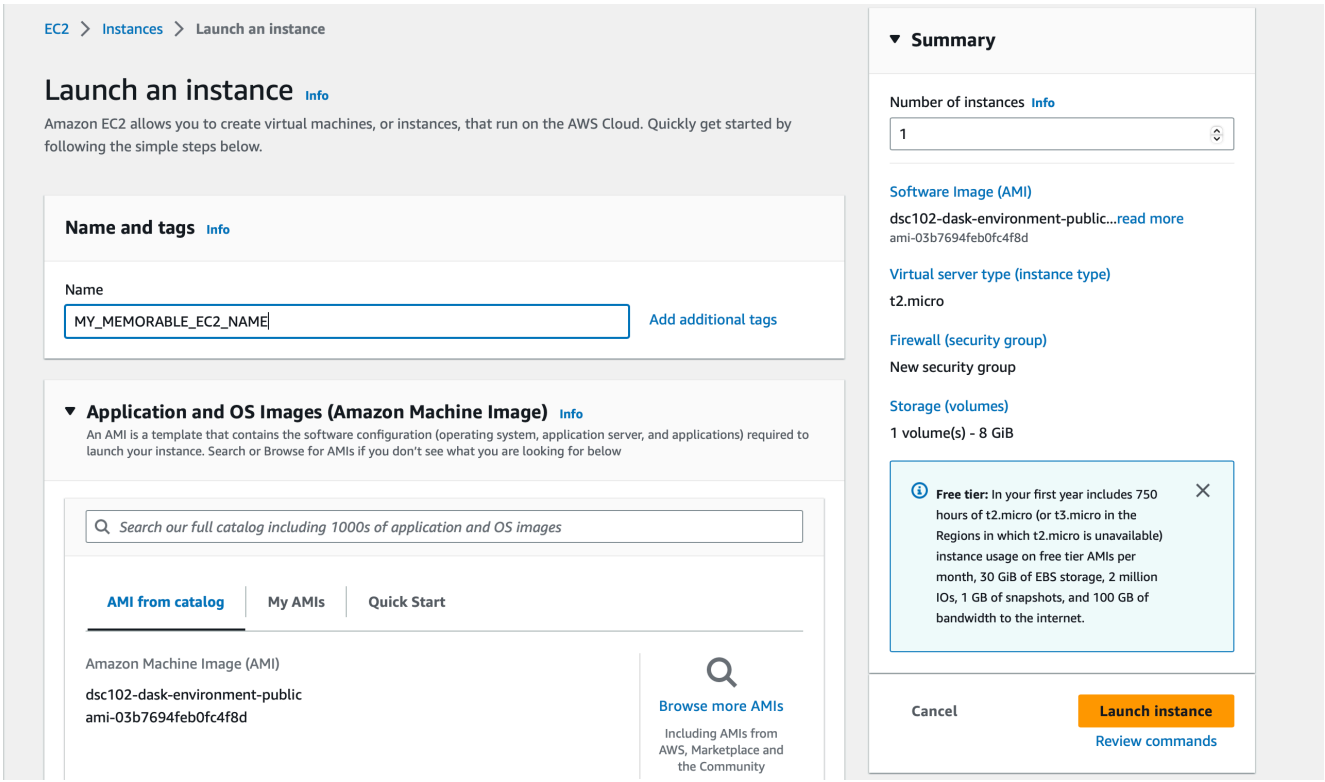


Figure 3

b) Leave the 'Application and OS Images (Amazon Machine Image)' field as is, as that was pre-populated by your selection to run from the dsc102 AMI. Under 'instance type', select "t2.xlarge". Under the key pair (login) heading, click 'create new key pair', give the key pair a name that you will remember, leave 'key pair type' RSA checked, and then select the private key file format '.pem' if your personal computer is Mac or Linux, or '.ppk' if you are using Putty on Windows. Once you have performed this step, you will only have to select your existing key pair for future iterations. Download the key to a location you will remember as you will be reusing this each time you want to log in to your machine. Here is a more info for mac users: `https://www.youtube.com/watch?v=8UqtMcX_kg0` and for Windows users: `https://www.youtube.com/watch?v=kzLRxVgos2M` Under the 'network settings' header, create a new security group and leave the "Allow SSH traffic from . . . Anywhere 0.0.0.0/0 checked".

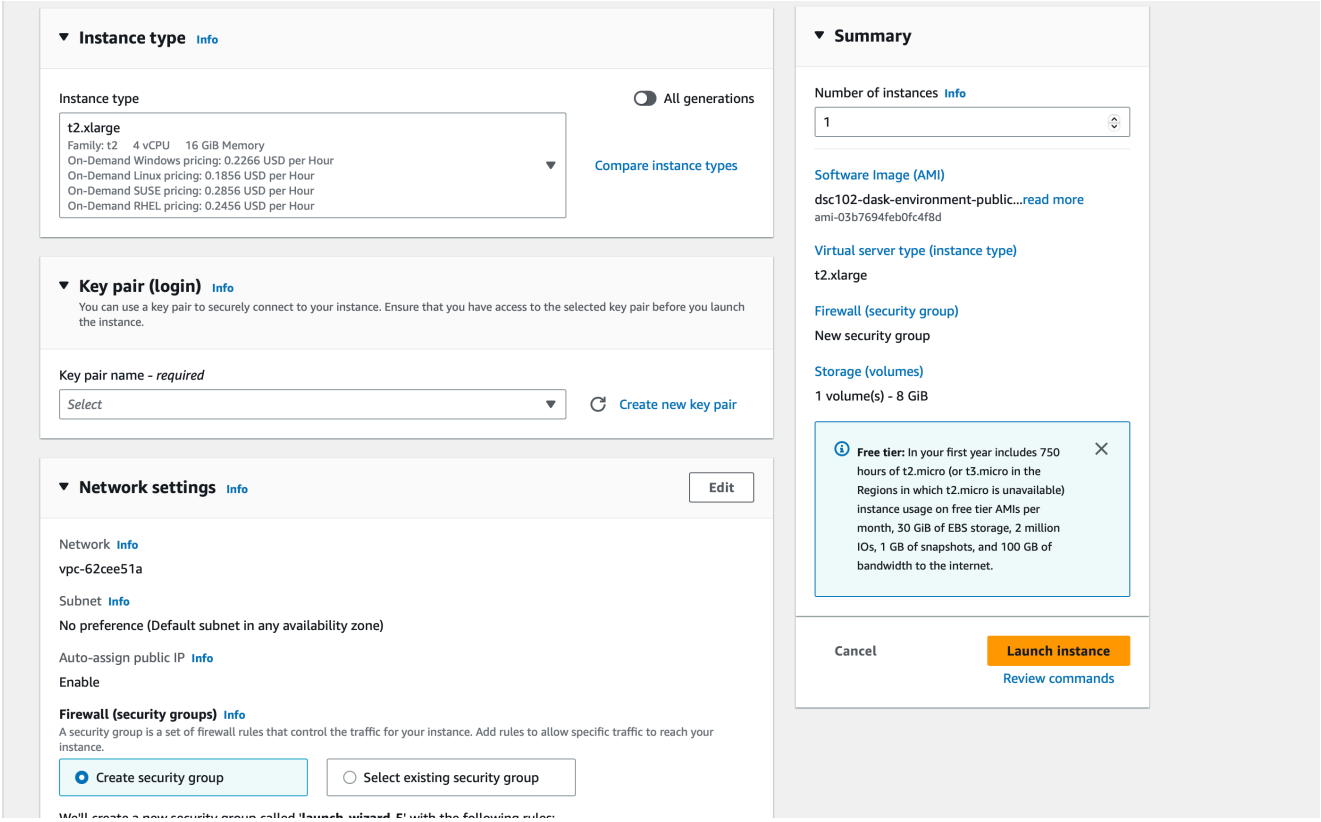

Instance type Info
Instance type
All generations
t2.xlarge
Family: t2 4 vCPU 16 GiB Memory
On-Demand Windows pricing: 0.2266 USD per Hour
On-Demand Linux pricing: 0.1856 USD per Hour
On-Demand SUSE pricing: 0.2856 USD per Hour
On-Demand RHEL pricing: 0.2456 USD per Hour
Compare instance types
Key pair (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
vpc-62cee51a
Subnet Info
No preference (Default subnet in any availability zone)
Auto-assign public IP Info
Enable
Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Create security group
Select existing security group
Summary
Number of instances Info
1
Software Image (AMI)
dsc102-dask-environment-public...read more
ami-03b7694feb0fc4f8d
Virtual server type (instance type)
t2.xlarge
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
Review commands

Figure 4

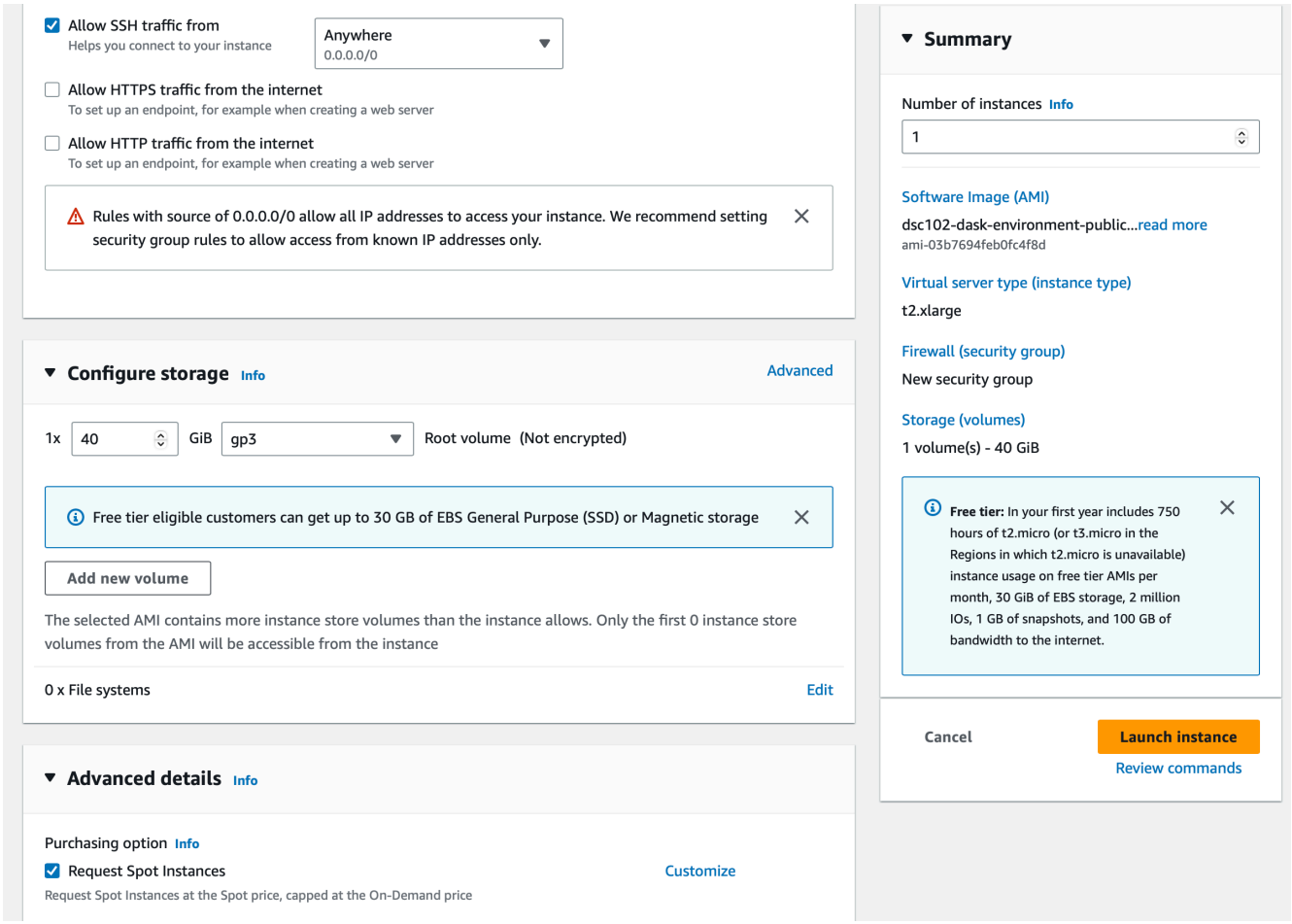


Figure 5

c) Under 'Configure Storage', select "40GB" of storage on a 'general purpose SSD (gp3)'. Under 'Advanced Details', check the 'Request Spot Instances' box and leave all the sub-values as defaults. Under 'IAM instance profile' select 'Dsc102Role_InstanceProfile.' NOTE: If you do not see 'InstanceProfile' in the list, leave it blank and note that you will have to manually authenticate on your nodes. See the 'Manual User Authentication' section below for details. Retain other fields unchanged.

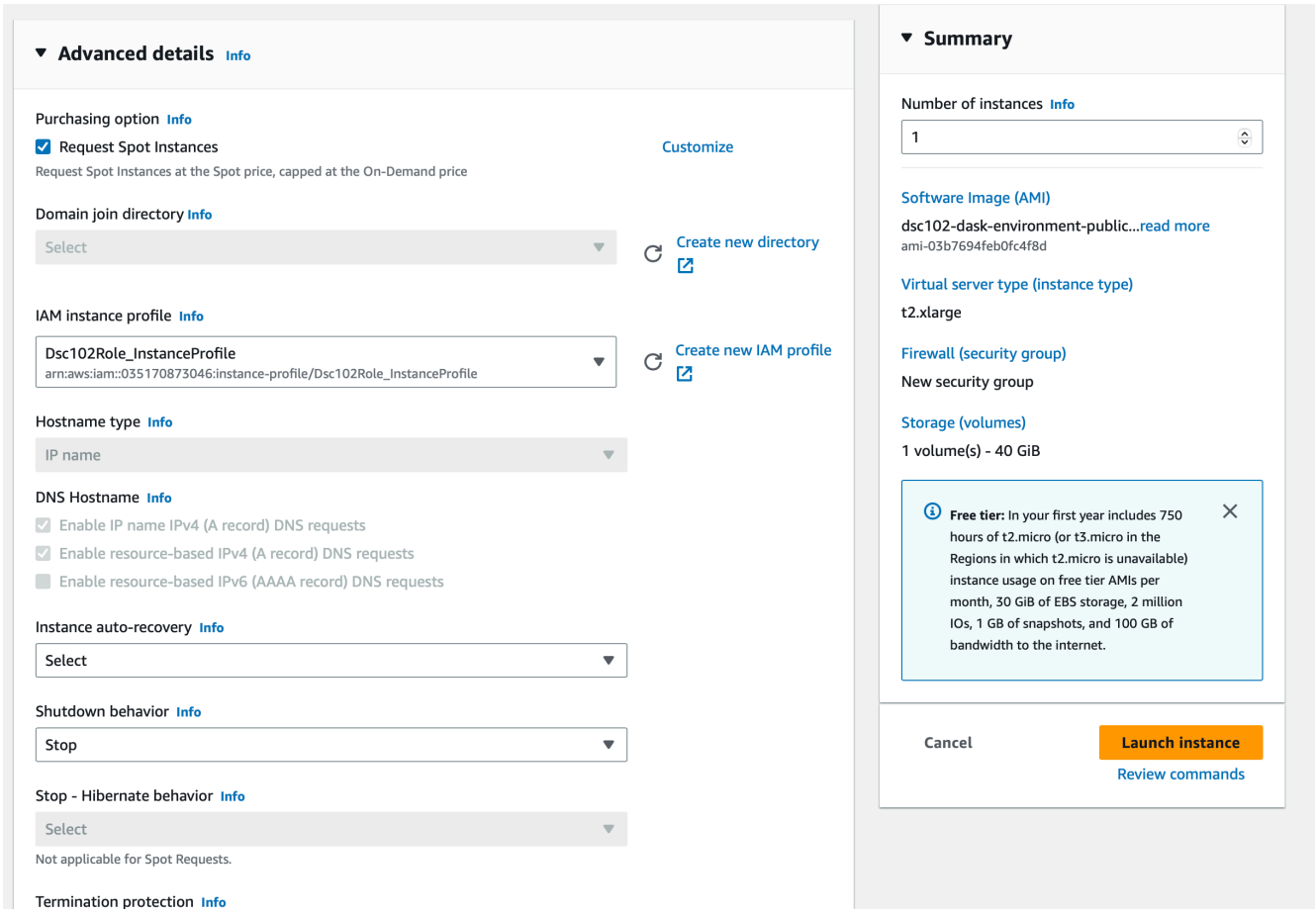


Figure 6

d) Finally, after pressing the "Launch Instance" button. Return to the 'Instances' page and wait for your instance's 'Instance state' to be set to 'Running'.

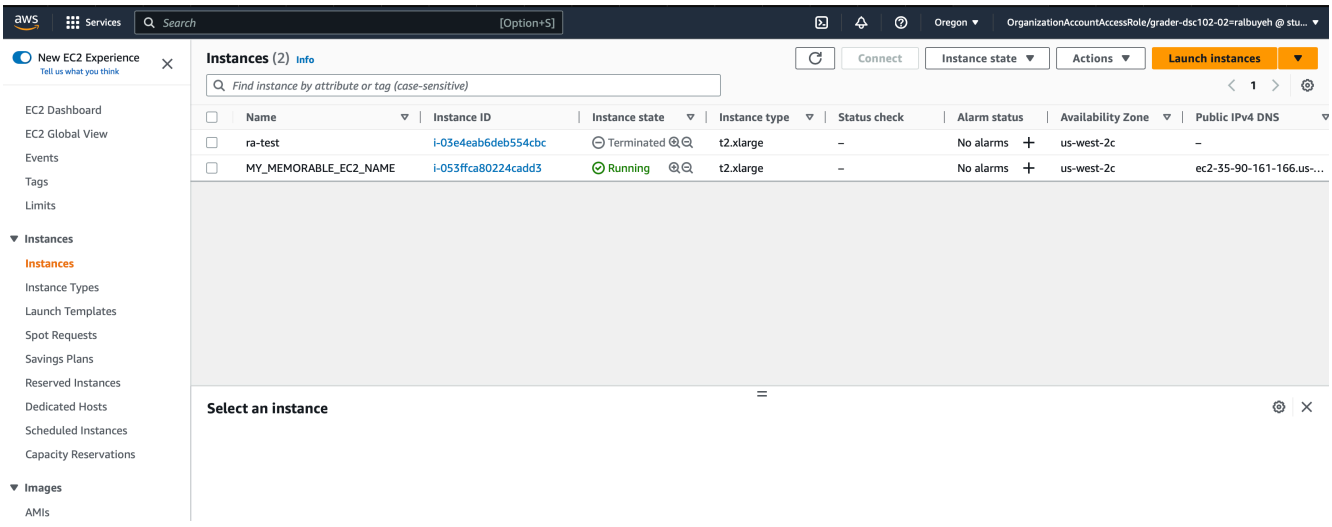


Figure 7

e) Click the instance ID and you should see details on your instance. Copy the public IPv4 addresss.

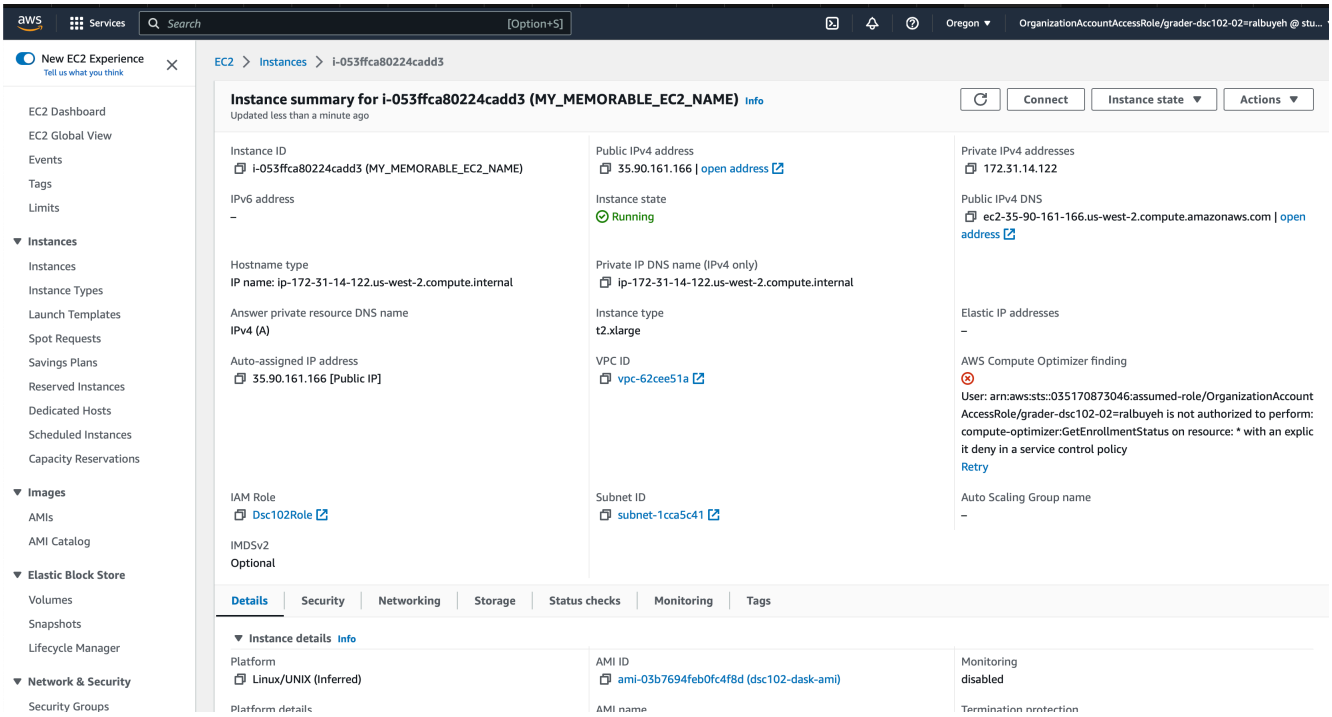


Figure 8

4) Next, you will start the jupyter notebook server on the instance.

a) Change permission of the ssh keyfile to make sure your private key file isn't publicly viewable: `chmod 400 <keyfilename>.pem`. Linux and Mac users in particular will need the chmod.

b) SSH into one of the nodes using command: `ssh -i ‘‘YOUR-KEY-NAME.pem’’ ubuntu@<ip-address-of-EC2-instance>`. This command is shown in the Figure 9 below. <ip-address-of-EC2-instance> is shown in the red box in Figure 10. Activate the dask environment with command: `source dask_env/bin/activate`. Start jupyter notebook server on one terminal with: `jupyter notebook --port=8888`.

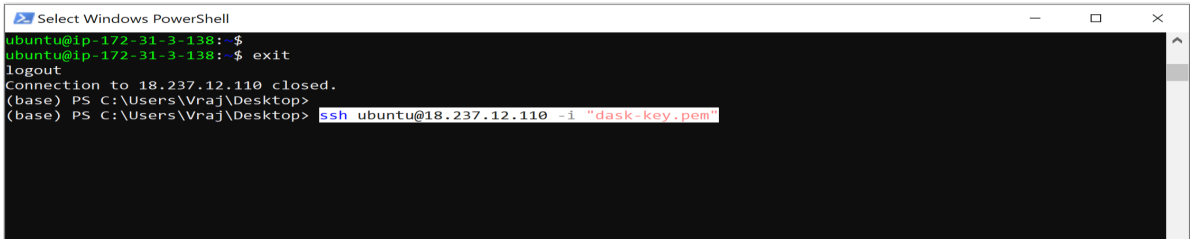


Figure 9

c) Open a new terminal and SSH to jupyter notebook using: `ssh -i ‘‘dask-key.pem’’ ubuntu@<ip-address-of-EC2-instance> -L 8888:localhost:8888`. '-L' will port forward any connection to port 8888 on the local machine to port 8888 on <ip-address-of-EC2-instance>. Run `source dask_env/bin/activate` again to re-activate the dask env in your terminal. Type in `jupyter notebook list` to get the token/password for the jupyter notebook. Open your browser and go to localhost:8888 and paste the token, or copy the entire path, as port 8888 is mapped to local. You can write your code here using jupyter notebook. To see dashboard on localhost port 8001 use command: `ssh -i ‘‘dask-key.pem’’ ubuntu@<ip-address-of-EC2-instance> -L 8001:localhost:8787`.

Consider using utilities like *tmux* or *nohup* for managing terminals.

5) The data and files are available from the s3 bucket (s3://dsc102-public). This contains the function signatures

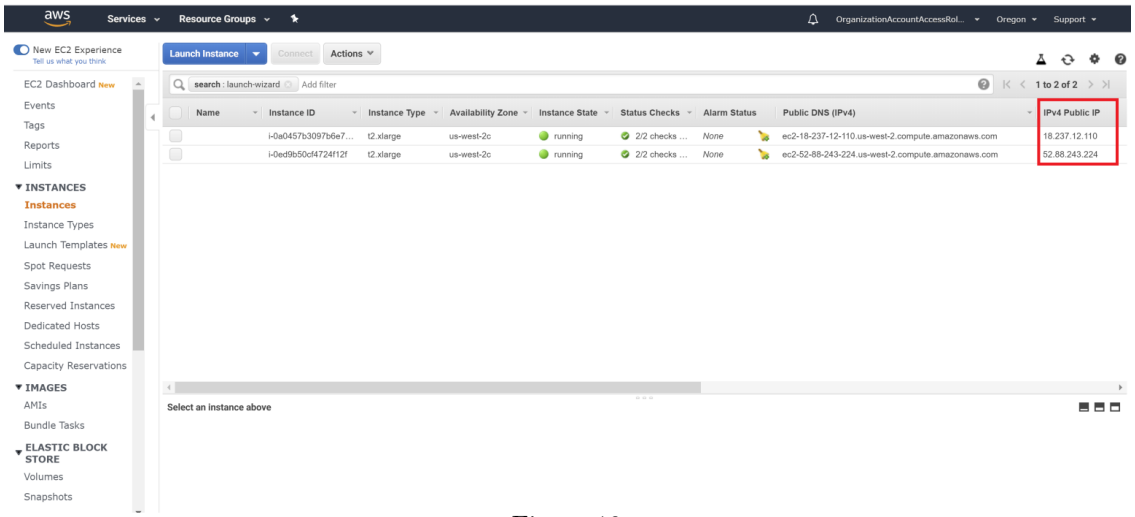


Figure 10

(PA0.py), dataset (user_reviews.csv), schema of expected output (OutputSchema_PA0.json), and the expected result on the development dataset (results_PA0.json).

((**Manual User Authentication**)) If you did not find the Dsc102Role_InstanceProfile as mentioned above when you were spinning up your EC2, you will need to manually authenticate on your EC2 instance using your user credentials. This means you are using your user permissions to access s3 rather than any permissions attached to the EC2 itself. Go to the UCSD ETS landing page where you clicked the link to access the AWS console. Instead of clicking 'Click here to access AWS,' click 'Generate API Keys (for CLI/scripting).' You will find three export statements there, corresponding to AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY, and AWS_SESSION_TOKEN. Copy all the text there into your EC2 terminal (where you just ssh-ed in), and you are now authenticated to copy objects from s3.

a) Verify that you have S3 access, specifically to our dsc102-public bucket, from your EC2 instance by running:

```
aws s3 ls s3://dsc102-public
```

You should see a listing of objects in the s3://dsc102-public bucket.

b) Use the command `aws s3 sync s3://dsc102-public /local-file-path` to download the files from S3 to local disk. Make sure that data is available in the same path where the jupyter notebook client is running.

6) Open the dashboard and click on "Workers" to double check if all workers (all threads of the single machine) are connected and you are now ready to code up.

7) Terminate the EC2 instance once you are done.

VERY IMPORTANT: Download your progress to your local machine (or backup to a private GitHub repo) at regular intervals and terminate your instance when you decide to pause working. You have only $50 for both PA0 and PA1 and so DO NOT leave instances running. If you terminate without downloading, you WILL LOSE all your work. Every time you start a new instance, you must download the dataset from S3 to your instance. Also, start only AWS Spot Instances and NOT On-Demand instances.