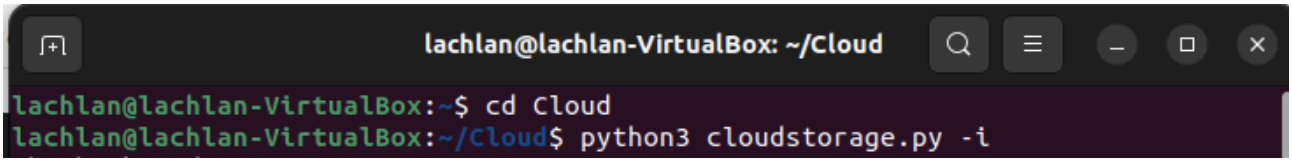
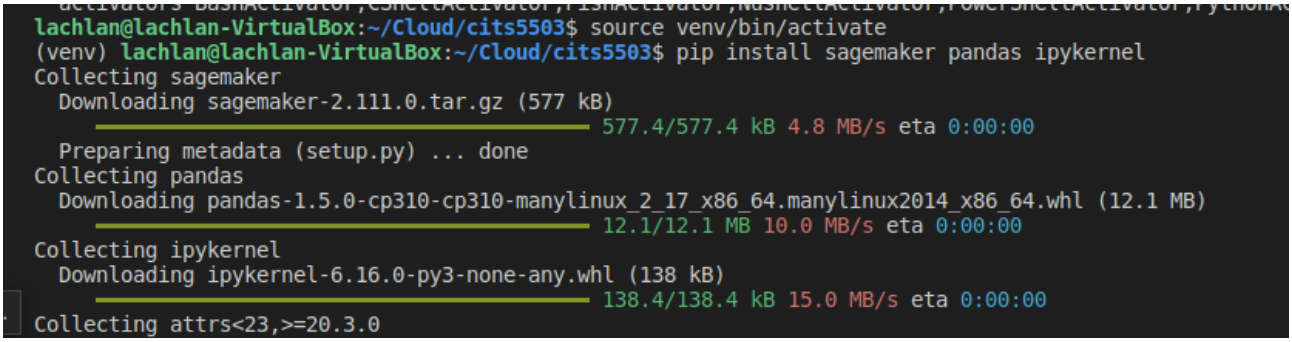
Creating Bucket



Activating Virtual environment and install needed packages.



Running the following code in VS CODE. (All outputs are included, some like this one had no output).

```
import sagemaker
import boto3

import numpy as np  # For matrix operations and numerical processing
import pandas as pd  # For munging tabular data
from time import gmtime, strftime
import os

region = 'ap-southeast-2'
smclient = boto3.Session().client("sagemaker")

iam = boto3.client('iam')
sagemaker_role = iam.get_role(RoleName='Role_AWS_SageMaker')['Role']['Arn']

student_id = "22975276"
bucket = '22975276-cloudstorage3'
prefix = f"sagemaker/{student_id}-hpo-xgboost-dm"
```

✓ 12.2s

```
!wget -N https://archive.ics.uci.edu/ml/machine-learning-databases/00222/bank-additional.zip
!unzip -o bank-additional.zip
```

✓ 5.6s

```
--2022-10-07 15:57:52--  https://archive.ics.uci.edu/ml/machine-learning-databases/00222/bank-additional.zip
Resolving archive.ics.uci.edu (archive.ics.uci.edu)... 128.195.10.252
Connecting to archive.ics.uci.edu (archive.ics.uci.edu)|128.195.10.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 444572 (434K) [application/x-httpd-php]
Saving to: 'bank-additional.zip'

bank-additional.zip 100%[===================>] 434.15K   313KB/s    in 1.4s

2022-10-07 15:57:56 (313 KB/s) - 'bank-additional.zip' saved [444572/444572]

Archive:  bank-additional.zip
   creating: bank-additional/
  inflating: bank-additional/.DS_Store
   creating: __MACOSX/
```

```
data = pd.read_csv("./bank-additional/bank-additional-full.csv", sep=";")
pd.set_option("display.max_columns", 500)  # Make sure we can see all of the columns
pd.set_option("display.max_rows", 50)  # Keep the output on one page
data
```

✓ 1.4s Python

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	duration	campaign	pdays	previ
0	56	housemaid	m	Screen Shot 2022-05-08 at 10.46.53 pm	no	no	no	telephone	may	mon	261	1	999	
1	57	services	married	high.school	unknown	no	no	telephone	may	mon	149	1	999	
2	37	services	married	high.school	no	yes	no	telephone	may	mon	226	1	999	
3	40	admin.	married	basic.6y	no	no	no	telephone	may	mon	151	1	999	
4	56	services	married	high.school	no	no	yes	telephone	may	mon	307	1	999	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
41183	73	retired	married	professional.course	no	yes	no	cellular	nov	fri	334	1	999	
41184	46	blue-collar	married	professional.course	no	no	no	cellular	nov	fri	383	1	999	
41185	56	retired	married	university.degree	no	yes	no	cellular	nov	fri	189	2	999	

Answer the following questions: - Which variables are categorical? - Which ones are numerical?

```
data["no_previous_contact"] = np.where(
    data["pdays"] == 999, 1, 0
)  # Indicator variable to capture when pdays takes a value of 999
data["not_working"] = np.where(
    np.in1d(data["job"], ["student", "retired", "unemployed"]), 1, 0
)  # Indicator for individuals not actively employed
model_data = pd.get_dummies(data)  # Convert categorical variables to sets of indicators
model_data
```

[4] ✓ 1.3s Python

	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed	no_previous_contact	not_w
0	56	261	1	999	0	1.1	93.994	-36.4	4.857	5191.0	1	
1	57	149	1	999	0	1.1	93.994	-36.4	4.857	5191.0	1	
2	37	226	1	999	0	1.1	93.994	-36.4	4.857	5191.0	1	
3	40	151	1	999	0	1.1	93.994	-36.4	4.857	5191.0	1	
4	56	307	1	999	0	1.1	93.994	-36.4	4.857	5191.0	1	

```
model_data = model_data.drop(
    ["duration", "emp.var.rate", "cons.price.idx", "cons.conf.idx", "euribor3m", "nr.employed"],
    axis=1,
)
```

✓ 0.4s Python

```
model_data
```

✓ 0.5s Python

	age	campaign	pdays	previous	no_previous_contact	not_working	job_admin.	job_blue-collar	job_entrepreneur	job_housemaid	job_managen
0	56	1	999	0	1	0	0	0	0	1	
1	57	1	999	0	1	0	0	0	0	0	
2	37	1	999	0	1	0	0	0	0	0	
3	40	1	999	0	1	0	1	0	0	0	
4	56	1	999	0	1	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	

```
train_data, validation_data, test_data = np.split(
    model_data.sample(frac=1, random_state=1729),
    [int(0.7 * len(model_data)), int(0.9 * len(model_data))],
)

pd.concat([train_data["y_yes"], train_data.drop(["y_no", "y_yes"], axis=1)], axis=1).to_csv(
    "train.csv", index=False, header=False
)
pd.concat(
    [validation_data["y_yes"], validation_data.drop(["y_no", "y_yes"], axis=1)], axis=1
).to_csv("validation.csv", index=False, header=False)
pd.concat([test_data["y_yes"], test_data.drop(["y_no", "y_yes"], axis=1)], axis=1).to_csv(
    "test.csv", index=False, header=False
)
```

[7] ✓ 2.9s

```
boto3.Session().resource("s3").Bucket(bucket).Object(
    os.path.join(prefix, "train/train.csv")
).upload_file("train.csv")
boto3.Session().resource("s3").Bucket(bucket).Object(
    os.path.join(prefix, "validation/validation.csv")
).upload_file("validation.csv")
```

✓ 9.6s

Setup Hyperparameter Optimization

```
from time import gmtime, strftime, sleep

# Names have to be unique. You will get an error if you reuse the same name
tuning_job_name = f"{student_id}-xgboost-tuningjob-01"

print(tuning_job_name)

tuning_job_config = {
    "ParameterRanges": {
        "CategoricalParameterRanges": [],
        "ContinuousParameterRanges": [
            {
                "MaxValue": "1",
                "MinValue": "0",
                "Name": "eta",
            },
            {
                "MaxValue": "10",
                "MinValue": "1",
                "Name": "min_child_weight",
            },
            {
                "MaxValue": "2",
                "MinValue": "0",
                "Name": "alpha",
            },
        ],
        "IntegerParameterRanges": [
            {
                "MaxValue": "10",
                "MinValue": "1",
                "Name": "max_depth",
            }
        ],
    },
    "ResourceLimits": {"MaxNumberOfTrainingJobs": 2, "MaxParallelTrainingJobs": 2},
    "Strategy": "Bayesian",
    "HyperParameterTuningJobObjective": {"MetricName": "validation:auc", "Type": "Maximize"},
}
```

[9] ✓ 0.6s

```
22975276-xgboost-tuningjob-01
```

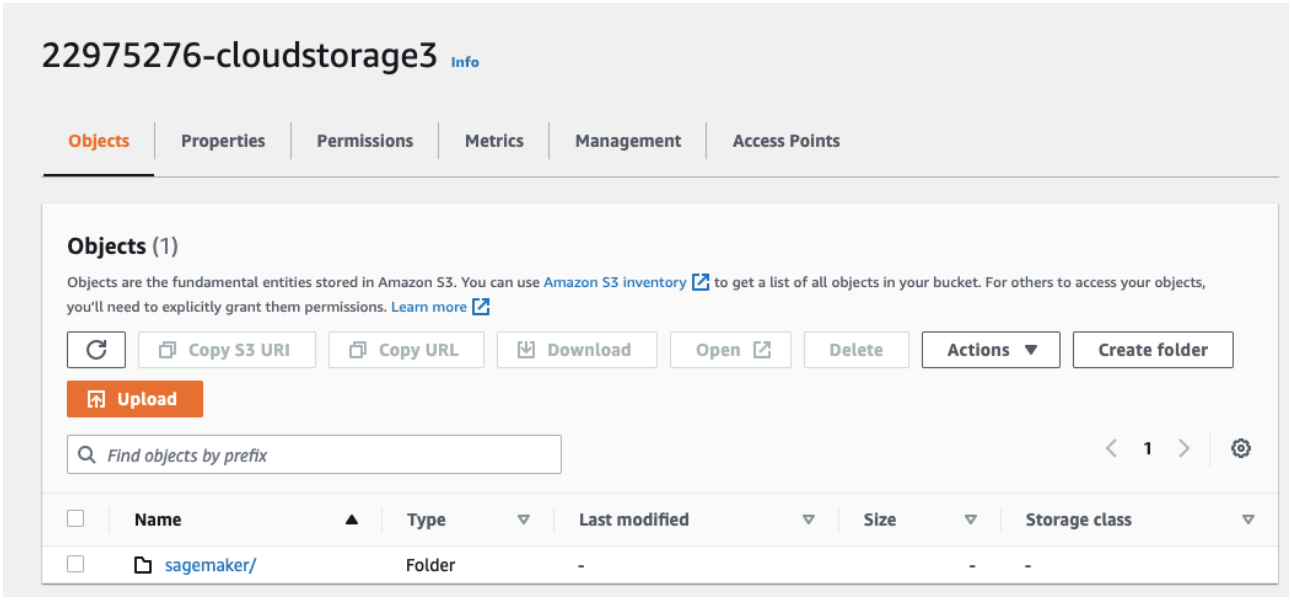
```python
from sagemaker.image_uris import retrieve
# Use XGBoost algorithm for training
training_image = retrieve(framework="xgboost", region=region, version="latest")

s3_input_train = "s3://{}/{}/train".format(bucket, prefix)
s3_input_validation = "s3://{}/{}/validation/".format(bucket, prefix)

training_job_definition = {
    "AlgorithmSpecification": {"TrainingImage": training_image, "TrainingInputMode": "File"},
    "InputDataConfig": [
        {
            "ChannelName": "train",
            "CompressionType": "None",
            "ContentType": "csv",
            "DataSource": {
                "S3DataSource": {
                    "S3DataDistributionType": "FullyReplicated",
                    "S3DataType": "S3Prefix",
                    "S3Uri": s3_input_train,
                }
            },
        },
        {
            "ChannelName": "validation",
            "CompressionType": "None",
            "ContentType": "csv",
            "DataSource": {
                "S3DataSource": {
                    "S3DataDistributionType": "FullyReplicated",
                    "S3DataType": "S3Prefix",
                    "S3Uri": s3_input_validation,
                }
            },
        },
    ],
    "OutputDataConfig": {"S3OutputPath": "s3://{}/{}/output".format(bucket, prefix)},
    "ResourceConfig": {"InstanceCount": 1, "InstanceType": "ml.m5.xlarge", "VolumeSizeInGB": 10},
    "RoleArn": sagemaker_role,
    "StaticHyperParameters": {
        "eval_metric": "auc",
        "num_round": "1",
        "objective": "binary:logistic",
        "rate_drop": "0.3",
        "tweedie_variance_power": "1.4",
    },
    "StoppingCondition": {"MaxRuntimeInSeconds": 43200},
}
```

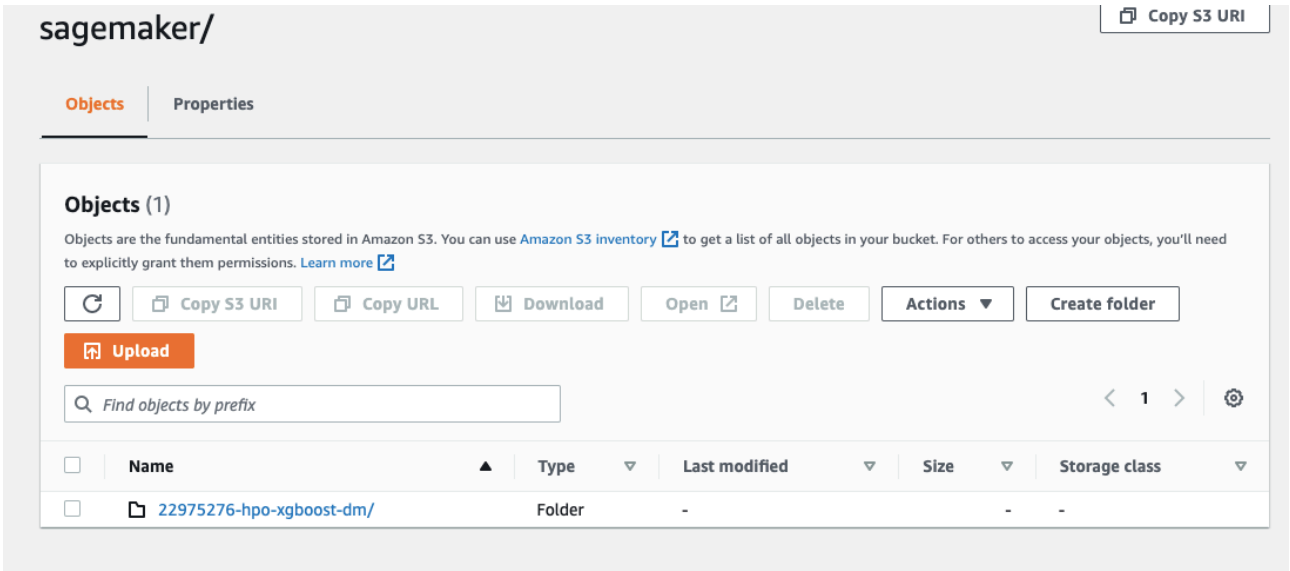
0.5s

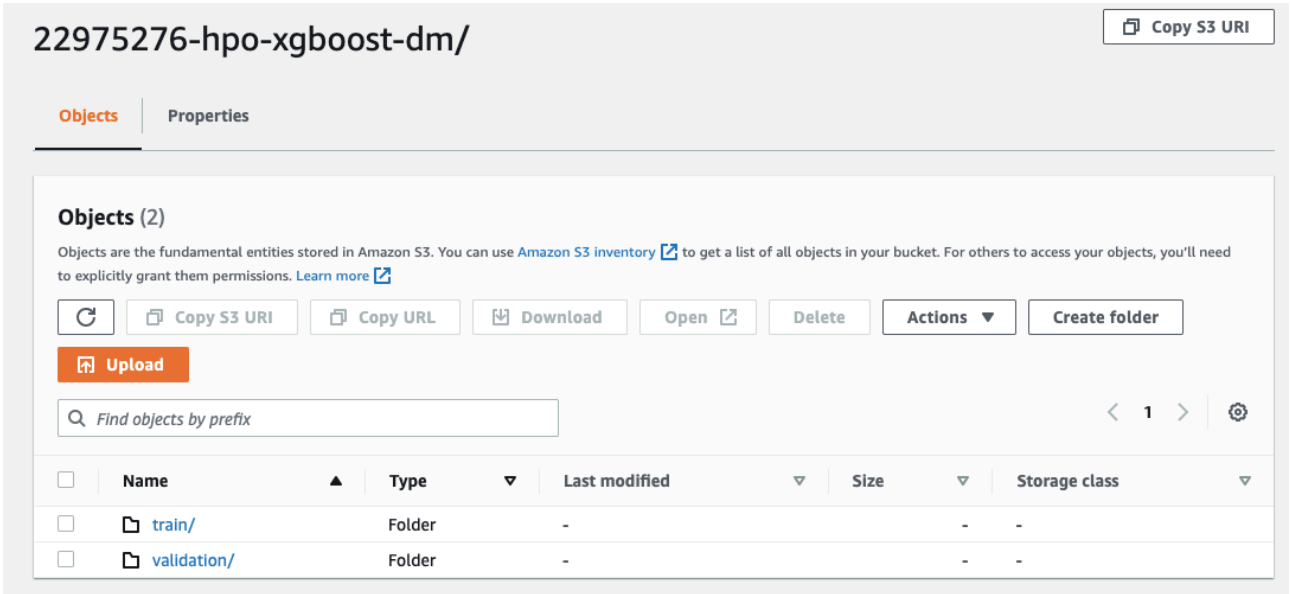
```python
#Launch Hyperparameter Tuning Job
smclient.create_hyper_parameter_tuning_job(
    HyperParameterTuningJobName=tuning_job_name,
    HyperParameterTuningJobConfig=tuning_job_config,
    TrainingJobDefinition=training_job_definition,
)
```

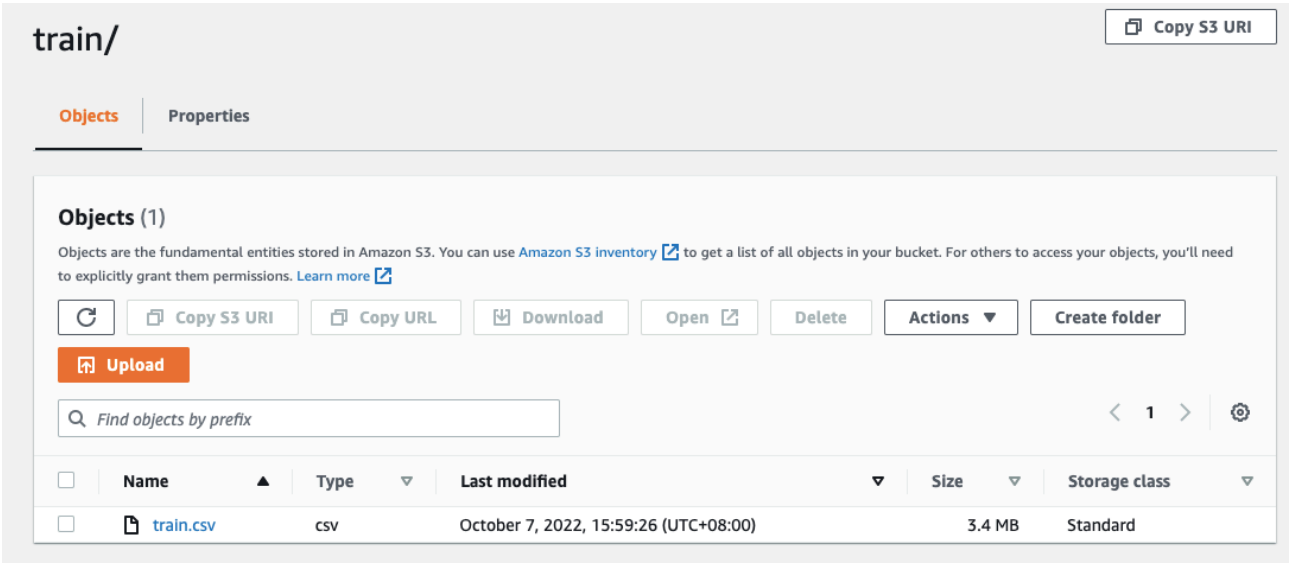
11] ⊗ 11.4s Pytho

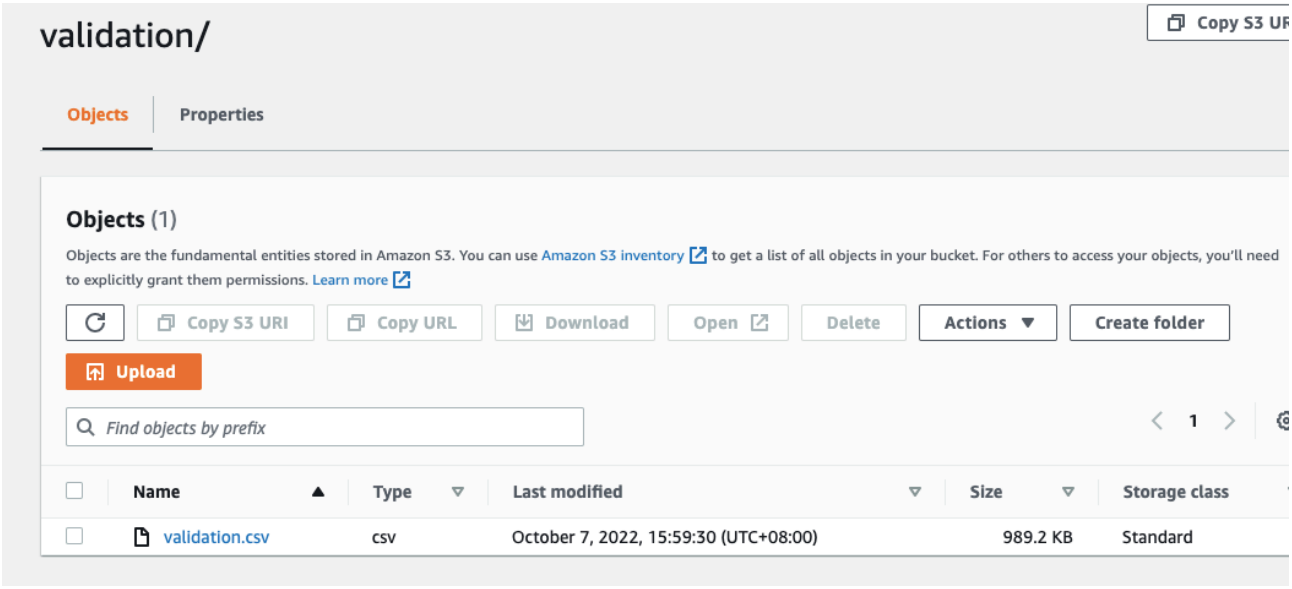
```
---------------------------------------------------------------------------
ResourceLimitExceeded                     Traceback (most recent call last)
Cell In [11], line 2
      1 #Launch Hyperparameter Tuning Job
----> 2 smclient.create_hyper_parameter_tuning_job(
      3     HyperParameterTuningJobName=tuning_job_name,
      4     HyperParameterTuningJobConfig=tuning_job_config,
      5     TrainingJobDefinition=training_job_definition,
      6 )

File ~/Cloud/cits5503/venv/lib/python3.10/site-packages/botocore/client.py:514, in ClientCreator._create_api_method.<locals
>._api_call(self, *args, **kwargs)
    510     raise TypeError(
    511         f"{py_operation_name}() only accepts keyword arguments."
    512     )
    513 # The "self" in this scope is referring to the BaseClient.
--> 514 return self._make_api_call(operation_name, kwargs)

File ~/Cloud/cits5503/venv/lib/python3.10/site-packages/botocore/client.py:938, in BaseClient._make_api_call(self, operatio
n_name, api_params)
```


22975276-cloudstorage3 Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Objects (1)
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
sagemaker/
Folder
-
-
-


sagemaker/
Copy S3 URI
Objects
Properties
Objects (1)
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
22975276-hpo-xgboost-dm/
Folder
-
-
-


22975276-hpo-xgboost-dm/
Copy S3 URI
Objects
Properties
Objects (2)
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
train/
Folder
-
-
-
validation/
Folder
-
-
-





Questions

a) In your S3 bucket, how many folders were created using the script (under the "{student_id}-hpo-xgboost-dm" folder)? List their name.

Two folders train and validation.

b) How many Hyperparameter tuning jobs were created using the script?

One

c) What metric was used in this script to evaluate the training results?

AUC: area under curve metric.

d) What strategy was used in the tuning job?

Bayesian