Lab 8 Notes

Student ID: 22994257
Name: Gaoyuan Zhang

[Step 1] Install missing libraries

```
(base) PS C:\Users\Administrator> pip3 install sagemaker pandas ipykernel
Requirement already satisfied: pandas in d:\environments\anaconda\lib\site-packages (1.3.5)
Requirement already satisfied: ipykernel in d:\environments\anaconda\lib\site-packages (5.1.2)
Collecting sagemaker
  Downloading sagemaker-2.110.0.tar.gz (576 kB)
     |████████████████████████████████| 576 kB 3.2 MB/s
Requirement already satisfied: boto3<2.0,>=1.20.21 in d:\environments\anaconda\lib\site-packages (from sagemaker) (1.
84)
Requirement already satisfied: google-pasta in d:\environments\anaconda\lib\site-packages (from sagemaker) (0.1.8)
Requirement already satisfied: numpy<2.0,>=1.9.0 in d:\environments\anaconda\lib\site-packages (from sagemaker) (1.21

Requirement already satisfied: protobuf<4.0,>=3.1 in d:\environments\anaconda\lib\site-packages (from sagemaker) (3.1
)
Requirement already satisfied: packaging>=20.0 in d:\environments\anaconda\lib\site-packages (from sagemaker) (21.3)
Requirement already satisfied: pandas in d:\environments\anaconda\lib\site-packages (1.3.5)
Requirement already satisfied: numpy<2.0,>=1.9.0 in d:\environments\anaconda\lib\site-packages (from sagemaker) (1.21

Requirement already satisfied: python-dateutil>=2.7.3 in d:\environments\anaconda\lib\site-packages (from pandas) (2.
)
Requirement already satisfied: pytz>=2017.3 in d:\environments\anaconda\lib\site-packages (from pandas) (2019.3)
Requirement already satisfied: traitlets>=4.1.0 in d:\environments\anaconda\lib\site-packages (from ipykernel) (4.3.3
Requirement already satisfied: tornado>=4.2 in d:\environments\anaconda\lib\site-packages (from ipykernel) (6.1)
Requirement already satisfied: jupyter-client in d:\environments\anaconda\lib\site-packages (from ipykernel) (5.3.3)
Requirement already satisfied: ipython>=5.0.0 in d:\environments\anaconda\lib\site-packages (from ipykernel) (7.8.0)
```

[Step 2] Deal with dataset on Jupyter Notebook

```
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 = "22994257"
bucket = '22994257-lab8'
prefix = f"sagemaker/{student_id}-hpo-xgboost-dm"
```

```
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
```

	age	job	marital	education	default	housing	loan	contact	month	day_of_week	duration	campaign	pdays	previous	poutcome
0	56	housemaid	married	basic.4y	no	no	no	telephone	may	mon	261	1	999	0	nonexistent
1	57	services	married	high.school	unknown	no	no	telephone	may	mon	149	1	999	0	nonexistent
2	37	services	married	high.school	no	yes	no	telephone	may	mon	226	1	999	0	nonexistent
3	40	admin.	married	basic.6y	no	no	no	telephone	may	mon	151	1	999	0	nonexistent
4	56	services	married	high.school	no	no	yes	telephone	may	mon	307	1	999	0	nonexistent
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

n [4]:
```
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
```

Out[4]:

	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed	no_previous_contact	not_working	job_
0	56	261	1	999	0	1.1	93.994	-36.4	4.857	5191.0	1	0	
1	57	149	1	999	0	1.1	93.994	-36.4	4.857	5191.0	1	0	
2	37	226	1	999	0	1.1	93.994	-36.4	4.857	5191.0	1	0	
3	40	151	1	999	0	1.1	93.994	-36.4	4.857	5191.0	1	0	
4	56	307	1	999	0	1.1	93.994	-36.4	4.857	5191.0	1	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
41183	73	334	1	999	0	-1.1	94.767	-50.8	1.028	4963.6	1	1	
41184	46	383	1	999	0	-1.1	94.767	-50.8	1.028	4963.6	1	0	
41185	56	189	2	999	0	-1.1	94.767	-50.8	1.028	4963.6	1	1	
41186	44	442	1	999	0	-1.1	94.767	-50.8	1.028	4963.6	1	0	
41187	74	239	3	999	1	-1.1	94.767	-50.8	1.028	4963.6	1	1	

41188 rows × 67 columns

In [5]:
```
model_data = model_data.drop(
    ["duration", "emp.var.rate", "cons.price.idx", "cons.conf.idx", "euribor3m", "nr.employed"],
    axis=1,
)
model_data
```

Out[5]:

	age	campaign	pdays	previous	no_previous_contact	not_working	job_admin.	job_blue-collar	job_entrepreneur	job_housemaid	job_management	job_retir
0	56	1	999	0	1	0	0	0	0	1	0	
1	57	1	999	0	1	0	0	0	0	0	0	
2	37	1	999	0	1	0	0	0	0	0	0	
3	40	1	999	0	1	0	1	0	0	0	0	
4	56	1	999	0	1	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	
41183	73	1	999	0	1	1	0	0	0	0	0	
41184	46	1	999	0	1	0	0	1	0	0	0	
41185	56	2	999	0	1	1	0	0	0	0	0	
41186	44	1	999	0	1	0	0	0	0	0	0	
41187	74	3	999	1	1	1	0	0	0	0	0	

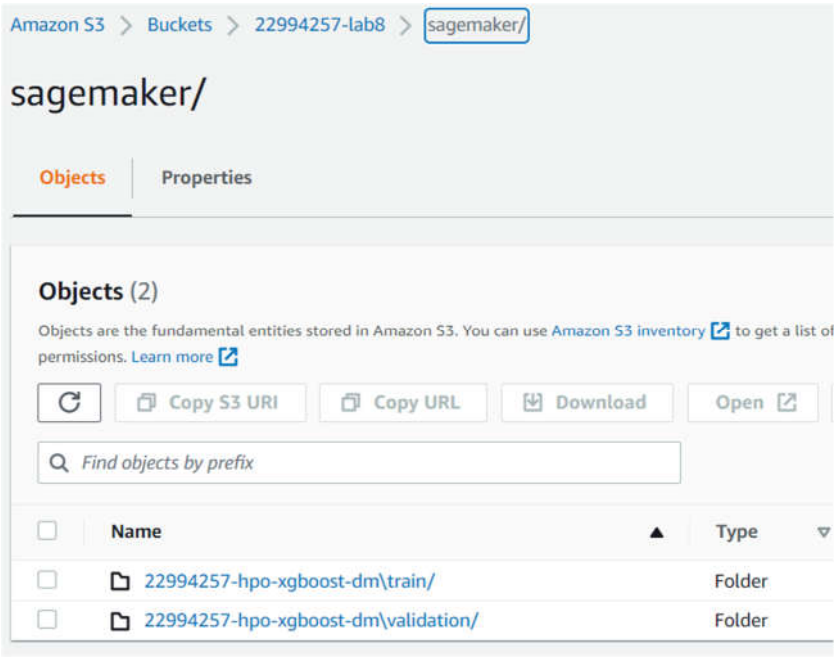
41188 rows × 61 columns

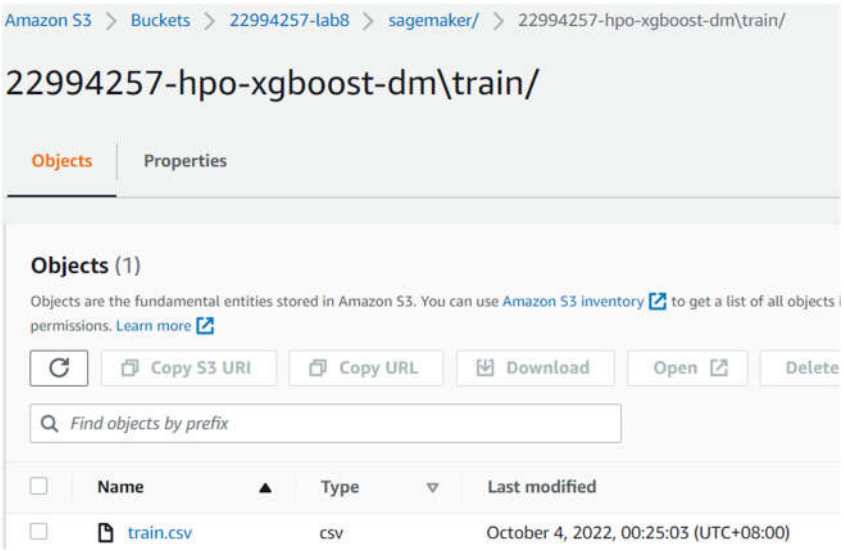
In [6]:
```
train_data, validation_data, test_data = np.split(
    model_data.sample(frac=1, random_state=1729),
    [int(0.7 * len(model_data)), int(0.9 * len(model_data))],
)

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
)
```

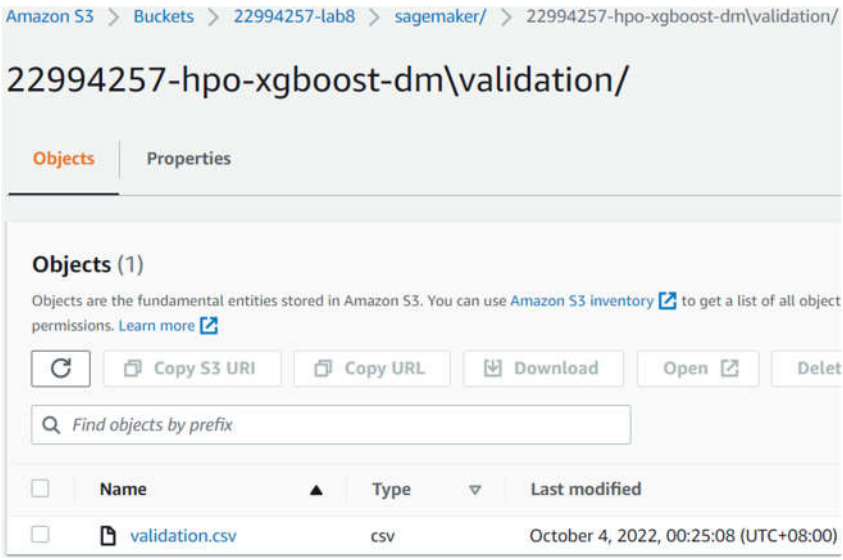
[Step 3] Create a S3 bucket and copy files

Create a bucket named “22994257-lab8” and then copy the dataset to my bucket.

In [7]:
```
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")
```


Amazon S3 > Buckets > 22994257-lab8 > sagemaker/
sagemaker/
Objects
Properties
Objects (2)
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of
permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Find objects by prefix
Name
Type
22994257-hpo-xgboost-dm\train/
Folder
22994257-hpo-xgboost-dm\validation/
Folder


Amazon S3 > Buckets > 22994257-lab8 > sagemaker/ > 22994257-hpo-xgboost-dm\train/
22994257-hpo-xgboost-dm\train/
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
permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Delete
Find objects by prefix
Name
Type
Last modified
train.csv
csv
October 4, 2022, 00:25:03 (UTC+08:00)


Amazon S3 > Buckets > 22994257-lab8 > sagemaker/ > 22994257-hpo-xgboost-dm\validation/
22994257-hpo-xgboost-dm\validation/
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 object
permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Delet
Find objects by prefix
Name
Type
Last modified
validation.csv
csv
October 4, 2022, 00:25:08 (UTC+08:00)

[Step 4] Setup Hyperparameter Optimization

In [8]:
```
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"},
}
```

```
22994257-xgboost-tuningjob-01
```

In [9]:
```
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",
    },
```

Finally, it can run but fail because of the resource limit on AWS.

In [10]:
```
#Launch Hyperparameter Tuning Job
smclient.create_hyper_parameter_tuning_job(
    HyperParameterTuningJobName=tuning_job_name,
    HyperParameterTuningJobConfig=tuning_job_config,
    TrainingJobDefinition=training_job_definition,
)
```

```
---------------------------------------------------------------------------
ResourceLimitExceeded                     Traceback (most recent call last)
<ipython-input-10-d8fa8d5f059d> in <module>
      3     HyperParameterTuningJobName=tuning_job_name,
      4     HyperParameterTuningJobConfig=tuning_job_config,
----> 5     TrainingJobDefinition=training_job_definition,
      6 )

D:\Environments\Anaconda\lib\site-packages\botocore\client.py in _api_call(self, *args, **kwargs)
    512                 )
    513             # The "self" in this scope is referring to the BaseClient.
--> 514             return self._make_api_call(operation_name, kwargs)
    515
    516         _api_call.__name__ = str(py_operation_name)

D:\Environments\Anaconda\lib\site-packages\botocore\client.py in _make_api_call(self, operation_name, api_params)
    936             error_code = parsed_response.get("Error", {}).get("Code")
    937             error_class = self.exceptions.from_code(error_code)
--> 938             raise error_class(parsed_response, operation_name)
    939         else:
    940             return parsed_response

ResourceLimitExceeded: An error occurred (ResourceLimitExceeded) when calling the CreateHyperParameterTuningJob operation: The account
-level service limit 'ml.m5.xlarge for training job usage' is 0 Instances, with current utilization of 0 Instances and a request delta o
f 2 Instances. Please contact AWS support to request an increase for this limit.
```

[Step 5] Answer the 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.
Answer: Three folders. Their name is “train” , “validation” and “output”.

b) How many Hyperparameter tuning jobs were created using the script?
Answer: Two.

```
"ResourceLimits": {"MaxNumberOfTrainingJobs": 2, "MaxParallelTrainingJobs": 2},
```

c) What metric was used in this script to evaluate the training results?
Answer: Auc(Area under the ROC Curve)

```
"StaticHyperParameters": {
    "eval_metric": "auc",
    "num_round": "1",
```

d) What strategy was used in the tuning job?
Answer: Bayesian network.

```
"Strategy": "Bayesian",
```