Final Exam - GuardRail and Shadow Test

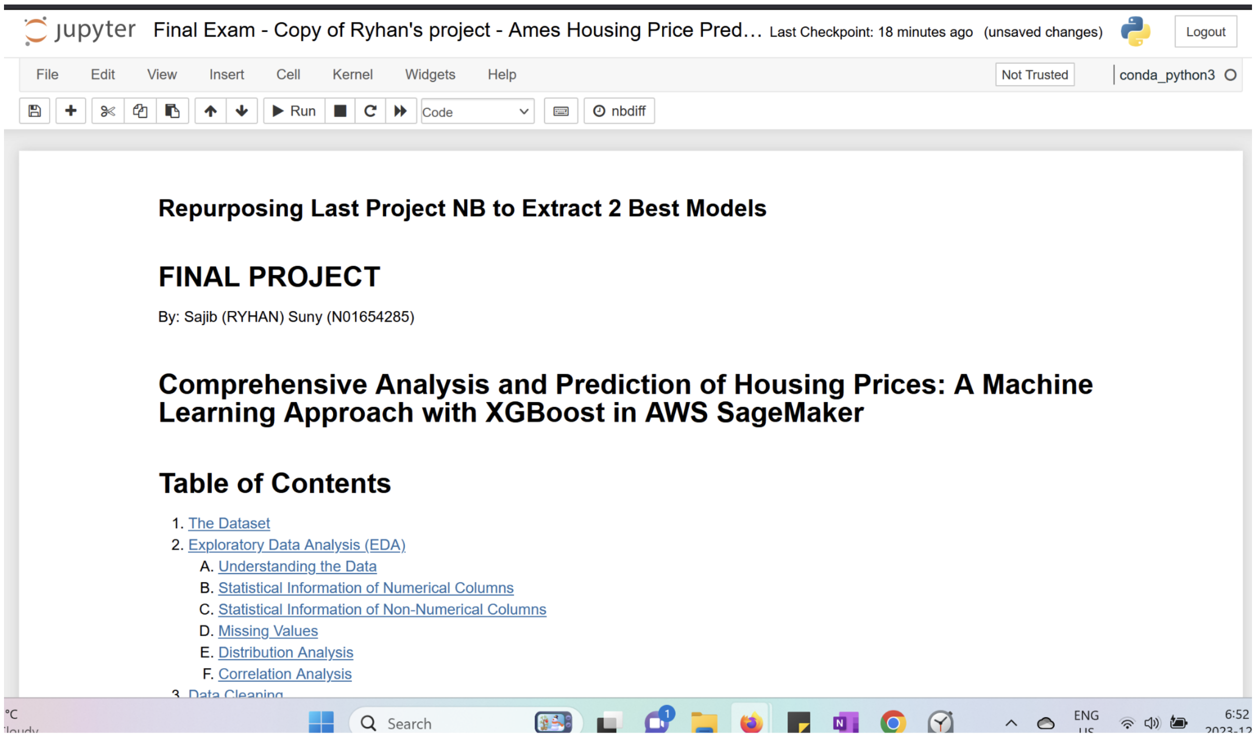
Sajib (RYHAN) Suny
N01654285

***** Prompt: You will simulate a deployment to production through SageMaker guardrail andSageMaker Shadow testing***
You will start from a data set and prepare the data for your project algorithm (2marks). You should show me the data set and algorithm you have used

1. GuardRail

## Showing DATASET and Algorithm

Source: Previous Project Notebook


jupyter
Final Exam - Copy of Ryhan's project - Ames Housing Price Pred… Last Checkpoint: 18 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
conda_python3
Run
Code
nbdiff
Search
ENG
6:52

Repurposing Last Project NB to Extract 2 Best Models

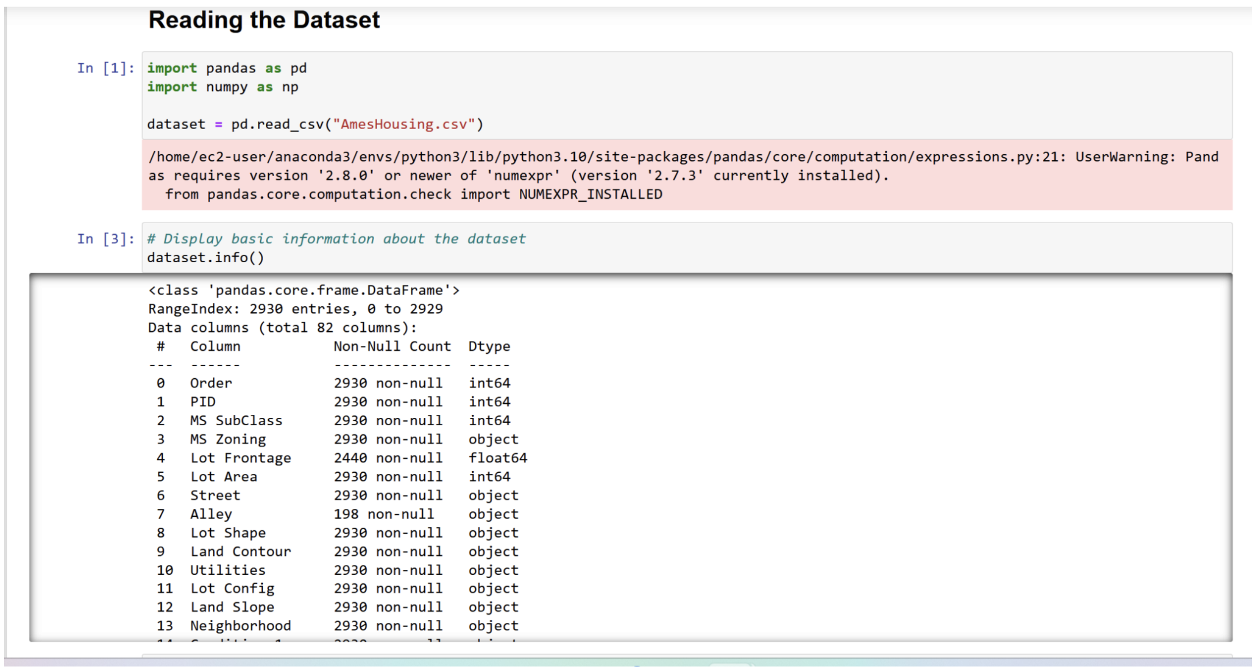
FINAL PROJECT

By: Sajib (RYHAN) Suny (N01654285)

Comprehensive Analysis and Prediction of Housing Prices: A Machine Learning Approach with XGBoost in AWS SageMaker

Table of Contents

- **Below Screenshot shows dataset - Ames housing prices**


Reading the Dataset

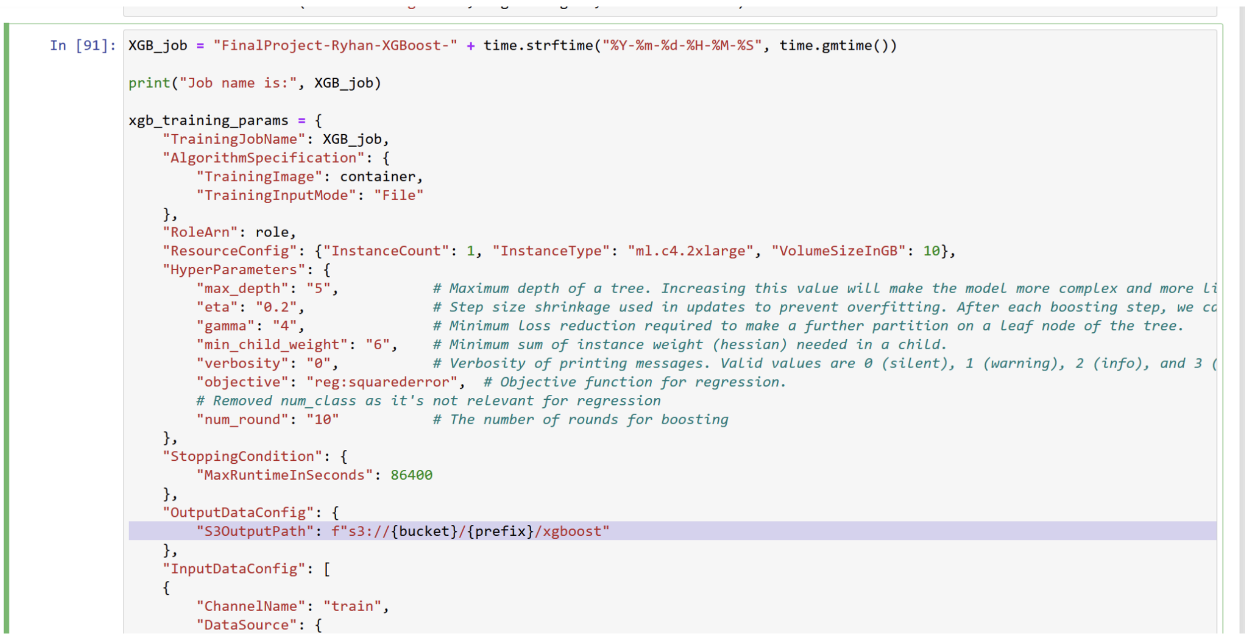
```
In [1]: import pandas as pd
        import numpy as np

        dataset = pd.read_csv("AmesHousing.csv")
```

```
/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/computation/expressions.py:21: UserWarning: Pand
as requires version '2.8.0' or newer of 'numexpr' (version '2.7.3' currently installed).
  from pandas.core.computation.check import NUMEXPR_INSTALLED
```

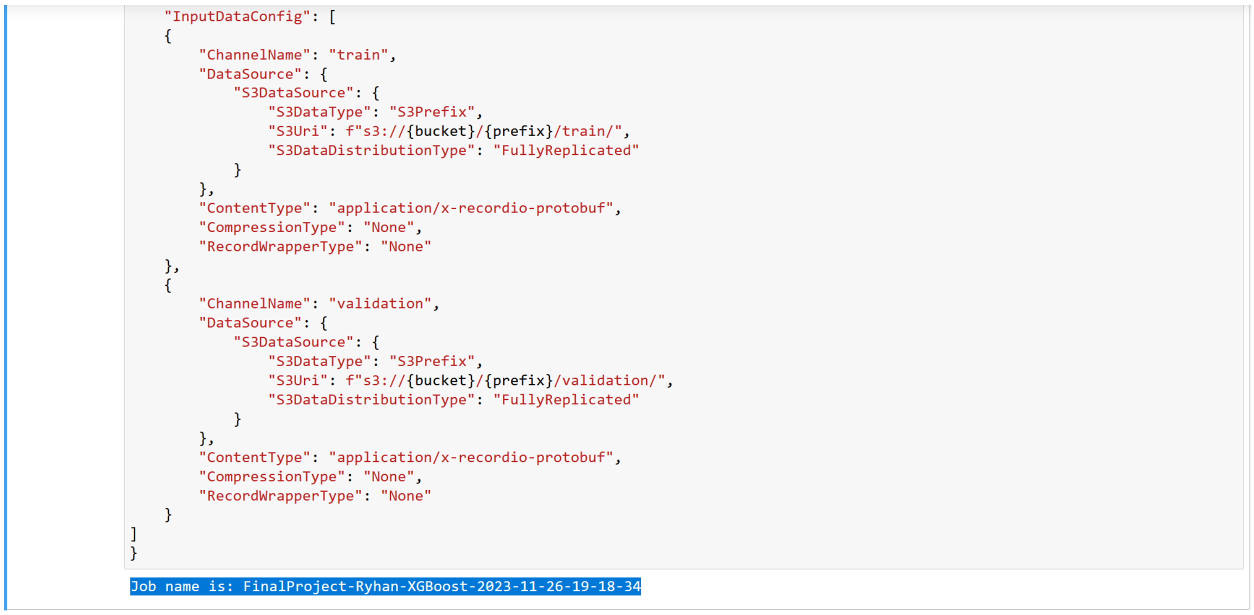
```
In [3]: # Display basic information about the dataset
        dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2930 entries, 0 to 2929
Data columns (total 82 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Order            2930 non-null   int64  
 1   PID              2930 non-null   int64  
 2   MS SubClass      2930 non-null   int64  
 3   MS Zoning        2930 non-null   object 
 4   Lot Frontage     2440 non-null   float64
 5   Lot Area         2930 non-null   int64  
 6   Street           2930 non-null   object 
 7   Alley            198 non-null    object 
 8   Lot Shape        2930 non-null   object 
 9   Land Contour     2930 non-null   object 
 10  Utilities        2930 non-null   object 
 11  Lot Config       2930 non-null   object 
 12  Land Slope       2930 non-null   object 
 13  Neighborhood     2930 non-null   object 
```

- Below screenshots show the normal Training Job configuration and job name, please note the last digits of the job name: ***18-34*** as an identifier.



```
In [91]: XGB_job = "FinalProject-Ryhan-XGBoost-" + time.strftime("%Y-%m-%d-%H-%M-%S", time.gmtime())

         print("Job name is:", XGB_job)

         xgb_training_params = {
             "TrainingJobName": XGB_job,
             "AlgorithmSpecification": {
                 "TrainingImage": container,
                 "TrainingInputMode": "File"
             },
             "RoleArn": role,
             "ResourceConfig": {"InstanceCount": 1, "InstanceType": "ml.c4.2xlarge", "VolumeSizeInGB": 10},
             "HyperParameters": {
                 "max_depth": "5",            # Maximum depth of a tree. Increasing this value will make the model more complex and more li
                 "eta": "0.2",                # Step size shrinkage used in updates to prevent overfitting. After each boosting step, we ca
                 "gamma": "4",                # Minimum loss reduction required to make a further partition on a leaf node of the tree.
                 "min_child_weight": "6",     # Minimum sum of instance weight (hessian) needed in a child.
                 "verbosity": "0",            # Verbosity of printing messages. Valid values are 0 (silent), 1 (warning), 2 (info), and 3 (
                 "objective": "reg:squarederror",  # Objective function for regression.
                 # Removed num_class as it's not relevant for regression
                 "num_round": "10"            # The number of rounds for boosting
             },
             "StoppingCondition": {
                 "MaxRuntimeInSeconds": 86400
             },
             "OutputDataConfig": {
                 "S3OutputPath": f"s3://{bucket}/{prefix}/xgboost"
             },
             "InputDataConfig": [
             {
                 "ChannelName": "train",
                 "DataSource": {
```



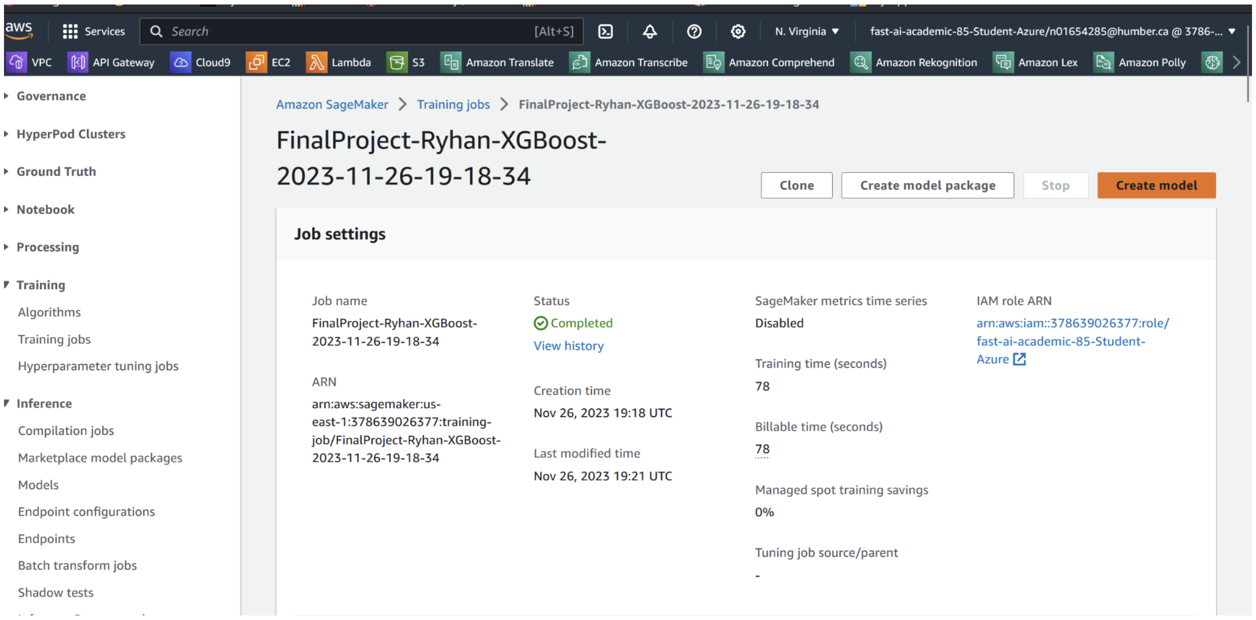
```
    "InputDataConfig": [
    {
        "ChannelName": "train",
        "DataSource": {
            "S3DataSource": {
                "S3DataType": "S3Prefix",
                "S3Uri": f"s3://{bucket}/{prefix}/train/",
                "S3DataDistributionType": "FullyReplicated"
            }
        },
        "ContentType": "application/x-recordio-protobuf",
        "CompressionType": "None",
        "RecordWrapperType": "None"
    },
    {
        "ChannelName": "validation",
        "DataSource": {
            "S3DataSource": {
                "S3DataType": "S3Prefix",
                "S3Uri": f"s3://{bucket}/{prefix}/validation/",
                "S3DataDistributionType": "FullyReplicated"
            }
        },
        "ContentType": "application/x-recordio-protobuf",
        "CompressionType": "None",
        "RecordWrapperType": "None"
    }
]
}

Job name is: FinalProject-Ryhan-XGBoost-2023-11-26-19-18-34
```

- Here is the normal Training Job with Xgboost Algorithm shown, job completed as shown in output and in GUI below

TRAINING

In [92]:
```
%%time

sm = boto3.client("sagemaker")

sm.create_training_job(**xgb_training_params)

# Checking training job status
status = sm.describe_training_job(TrainingJobName=XGB_job)["TrainingJobStatus"]
print(status)

sm.get_waiter("training_job_completed_or_stopped").wait(TrainingJobName=XGB_job)
if status == "Failed":
    message = sm.describe_training_job(TrainingJobName=XGB_job)["FailureReason"]
    print("Training failed with the following error: {}".format(message))
    raise Exception("Training job failed")
```

```
InProgress
CPU times: user 91.1 ms, sys: 5.51 ms, total: 96.6 ms
Wall time: 4min
```



FinalProject-Ryhan-XGBoost-2023-11-26-19-18-34

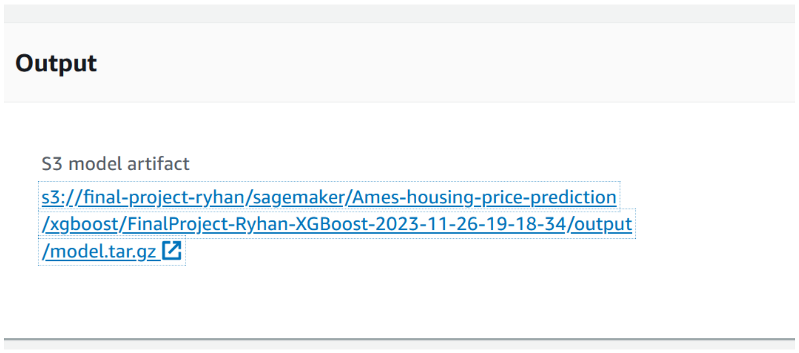
Job settings

Job name	Status	SageMaker metrics time series	IAM role ARN
FinalProject-Ryhan-XGBoost-2023-11-26-19-18-34	Completed	Disabled	arn:aws:iam::378639026377:role/fast-ai-academic-85-Student-Azure
	View history		
ARN	Creation time	Training time (seconds)	
arn:aws:sagemaker:us-east-1:378639026377:training-job/FinalProject-Ryhan-XGBoost-2023-11-26-19-18-34	Nov 26, 2023 19:18 UTC	78	
	Last modified time	Billable time (seconds)	
	Nov 26, 2023 19:21 UTC	78	
		Managed spot training savings	
		0%	
		Tuning job source/parent	
		-	

__**NEXT Prompt: You train two models. Both model artifacts have to be prepared in notebook. Youcan use SageMaker Hyperparameter tuning job to create a set of models and youpick the best two (Do not use web console). Make sure one model has a bettermetric than the other one (4 marks)__

# Collecting model 1

- And from the normal Training Job I collect the first model (to serve as production model)



Output

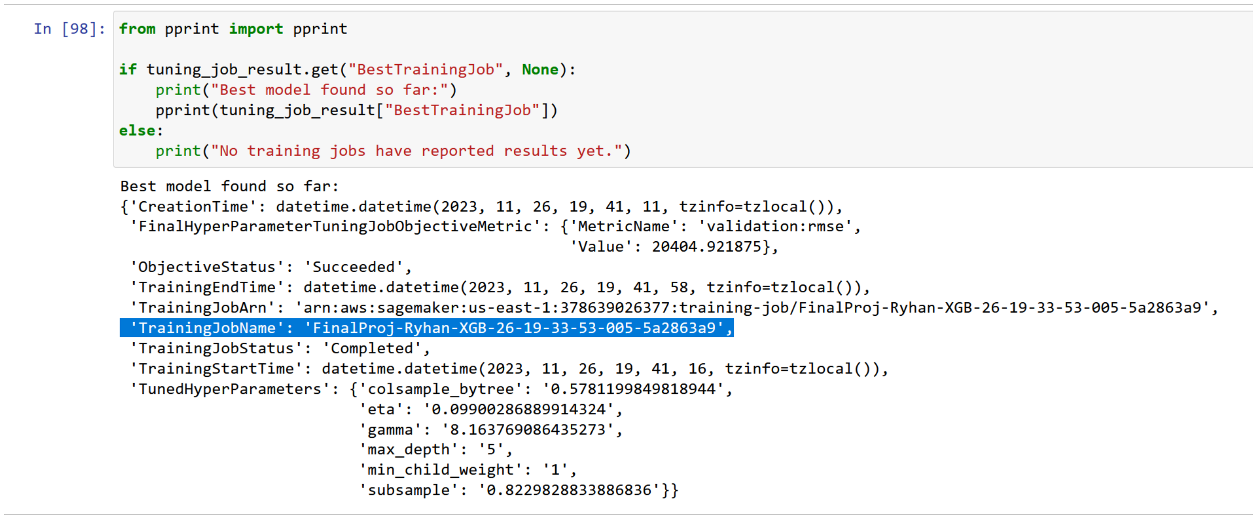
S3 model artifact
s3://final-project-ryhan/sagemaker/Ames-housing-price-prediction/xgboost/FinalProject-Ryhan-XGBoost-2023-11-26-19-18-34/output/model.tar.gz

shown below:

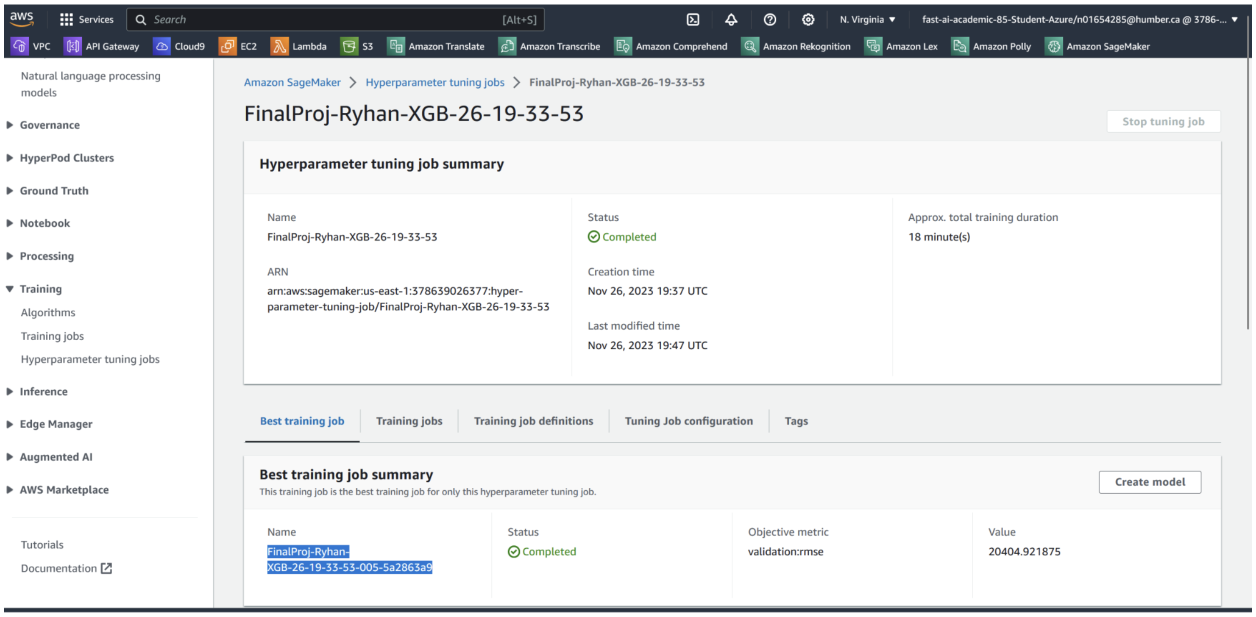
->PROD Second Best Model Artifact = Name Ending With *__18-34__*

# Collecting model 2

- Below shown Tuning Job Completed and the best training job name ends with ***__3a9__***



```
In [98]: from pprint import pprint

         if tuning_job_result.get("BestTrainingJob", None):
             print("Best model found so far:")
             pprint(tuning_job_result["BestTrainingJob"])
         else:
             print("No training jobs have reported results yet.")

         Best model found so far:
         {'CreationTime': datetime.datetime(2023, 11, 26, 19, 41, 11, tzinfo=tzlocal()),
          'FinalHyperParameterTuningJobObjectiveMetric': {'MetricName': 'validation:rmse',
                                                          'Value': 20404.921875},
          'ObjectiveStatus': 'Succeeded',
          'TrainingEndTime': datetime.datetime(2023, 11, 26, 19, 41, 58, tzinfo=tzlocal()),
          'TrainingJobArn': 'arn:aws:sagemaker:us-east-1:378639026377:training-job/FinalProj-Ryhan-XGB-26-19-33-53-005-5a2863a9',
          'TrainingJobName': 'FinalProj-Ryhan-XGB-26-19-33-53-005-5a2863a9',
          'TrainingJobStatus': 'Completed',
          'TrainingStartTime': datetime.datetime(2023, 11, 26, 19, 41, 16, tzinfo=tzlocal()),
          'TunedHyperParameters': {'colsample_bytree': '0.5781199849818944',
                                   'eta': '0.099002868889914324',
                                   'gamma': '8.163769086435273',
                                   'max_depth': '5',
                                   'min_child_weight': '1',
                                   'subsample': '0.8229828833886836'}}
```



FinalProj-Ryhan-XGB-26-19-33-53

Hyperparameter tuning job summary

Name
FinalProj-Ryhan-XGB-26-19-33-53

ARN
arn:aws:sagemaker:us-east-1:378639026377:hyper-parameter-tuning-job/FinalProj-Ryhan-XGB-26-19-33-53

Status
Completed

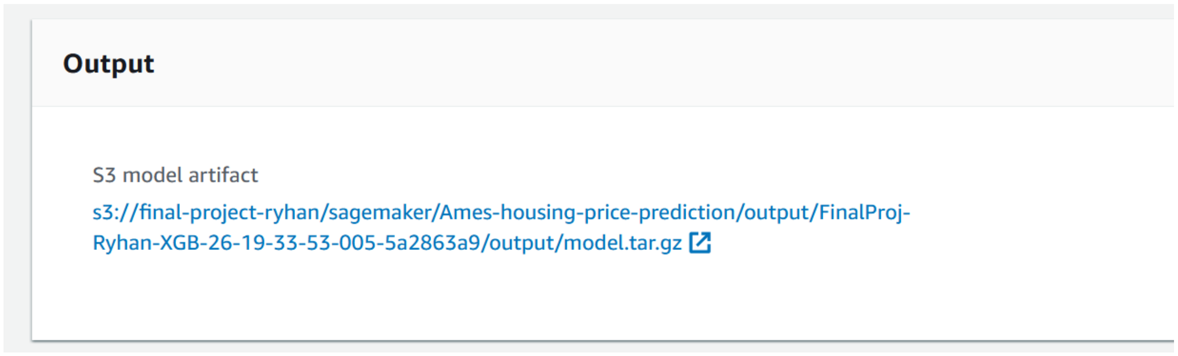
Creation time
Nov 26, 2023 19:37 UTC

Last modified time
Nov 26, 2023 19:47 UTC

Approx. total training duration
18 minute(s)

Best training job summary
This training job is the best training job for only this hyperparameter tuning job.

Name	Status	Objective metric	Value
FinalProj-Ryhan-XGB-26-19-33-53-005-5a2863a9	Completed	validation:rmse	20404.921875



Output

S3 model artifact
s3://final-project-ryhan/sagemaker/Ames-housing-price-prediction/output/FinalProj-Ryhan-XGB-26-19-33-53-005-5a2863a9/output/model.tar.gz

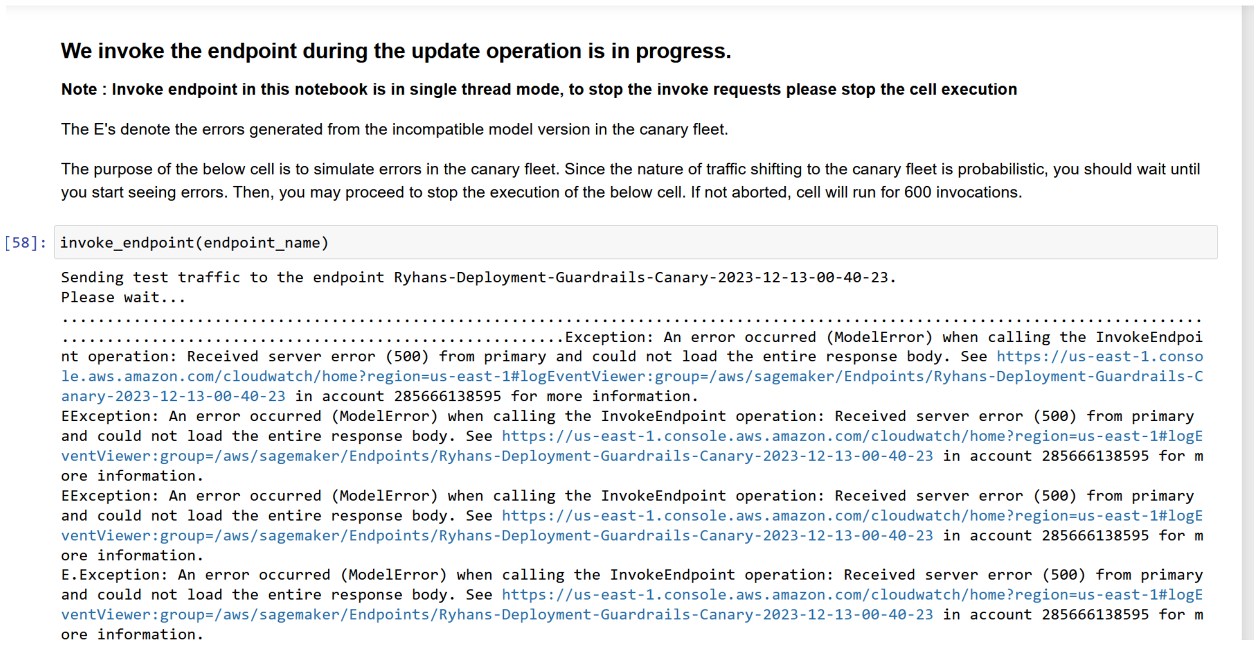
-> Improved Best Model Artifact = Ending With *3a9*

# Since this is the best model retrieved through HP Tuning, this model is better than previous normal Training Job

-> Please note: Until this part (collecting models from last project) I used MYAPPS, since last project was on myapps. Moving forward from here I use AWS ACADEMY as instructed.

*****NEXT Prompt: Use the notebook I gave as a starting point for simulating a guardrail. You areexpected to use parts of that code in your newly created notebook.• The first guardrail is a failed deployment that the guardrail rollbacksautomatically (5 marks). You must show me the "E" letters in the notebook.***

# GUARDRAIL FAILURE CASE



We invoke the endpoint during the update operation is in progress.

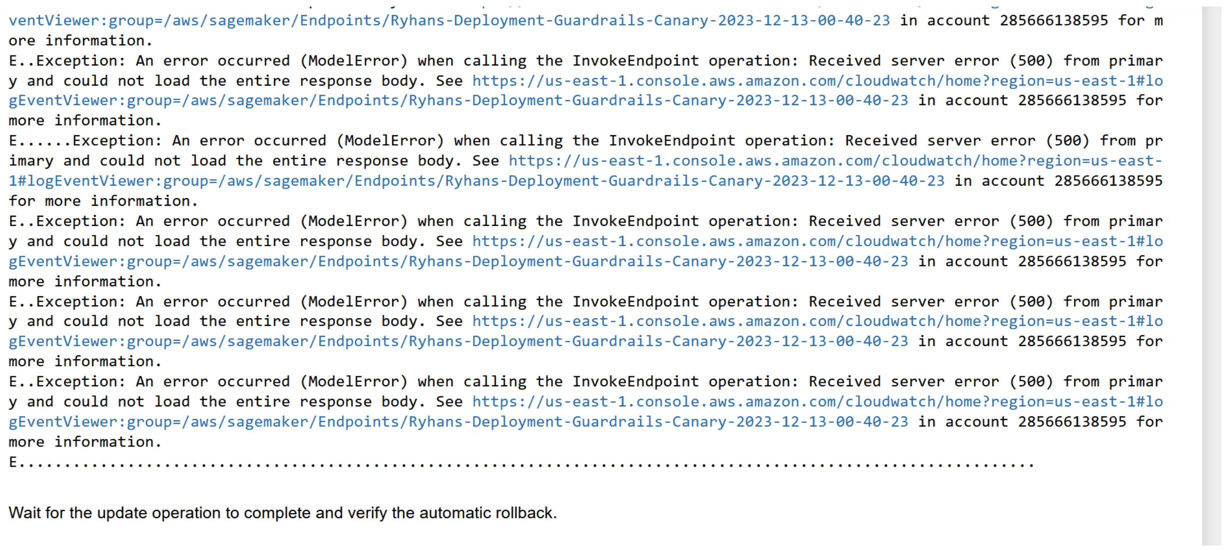
Note : Invoke endpoint in this notebook is in single thread mode, to stop the invoke requests please stop the cell execution

The E's denote the errors generated from the incompatible model version in the canary fleet.

The purpose of the below cell is to simulate errors in the canary fleet. Since the nature of traffic shifting to the canary fleet is probabilistic, you should wait until you start seeing errors. Then, you may proceed to stop the execution of the below cell. If not aborted, cell will run for 600 invocations.

```
[58]: invoke_endpoint(endpoint_name)
```

```
Sending test traffic to the endpoint Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23.
Please wait...
.........................................................................................................................
.....................................................Exception: An error occurred (ModelError) when calling the InvokeEndpoi
nt operation: Received server error (500) from primary and could not load the entire response body. See https://us-east-1.conso
le.aws.amazon.com/cloudwatch/home?region=us-east-1#logEventViewer:group=/aws/sagemaker/Endpoints/Ryhans-Deployment-Guardrails-C
anary-2023-12-13-00-40-23 in account 285666138595 for more information.
EException: An error occurred (ModelError) when calling the InvokeEndpoint operation: Received server error (500) from primary
and could not load the entire response body. See https://us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#logE
ventViewer:group=/aws/sagemaker/Endpoints/Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23 in account 285666138595 for m
ore information.
EException: An error occurred (ModelError) when calling the InvokeEndpoint operation: Received server error (500) from primary
and could not load the entire response body. See https://us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#logE
ventViewer:group=/aws/sagemaker/Endpoints/Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23 in account 285666138595 for m
ore information.
E.Exception: An error occurred (ModelError) when calling the InvokeEndpoint operation: Received server error (500) from primary
and could not load the entire response body. See https://us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#logE
ventViewer:group=/aws/sagemaker/Endpoints/Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23 in account 285666138595 for m
ore information.
```



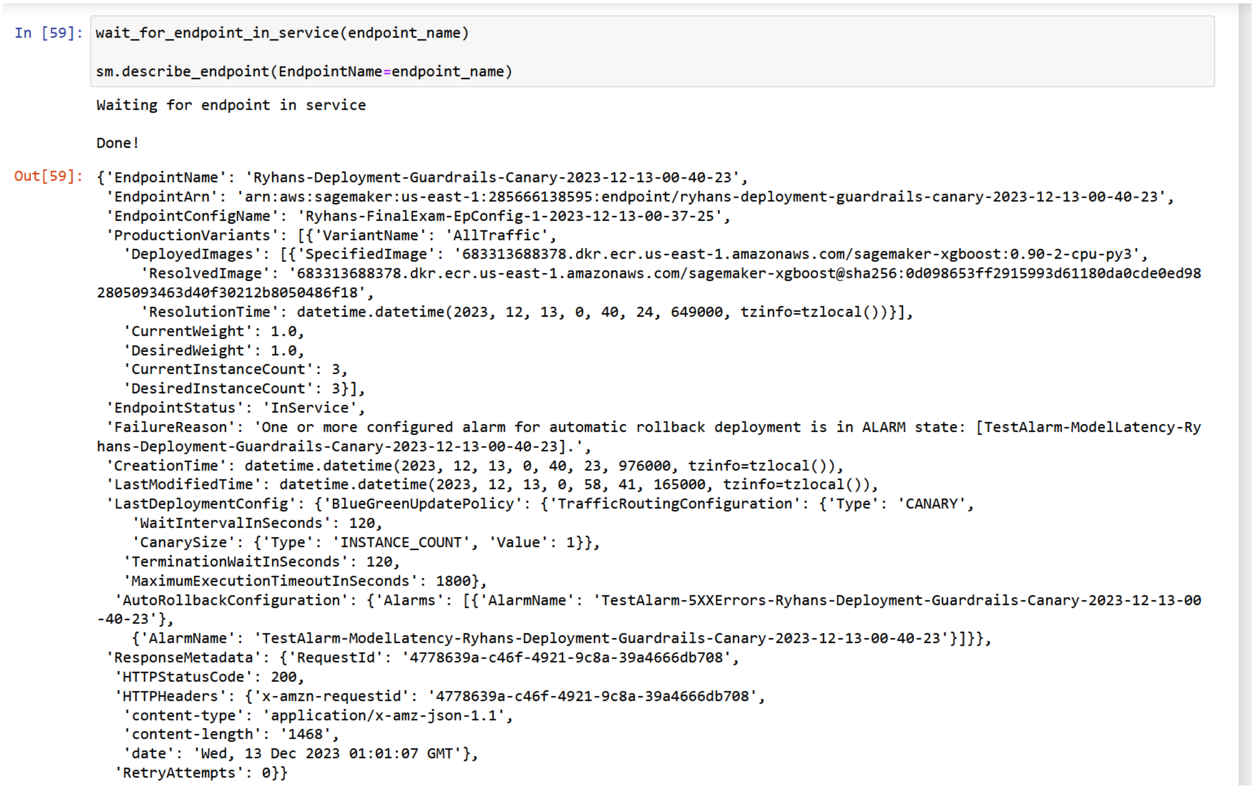
```
ventViewer:group=/aws/sagemaker/Endpoints/Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23 in account 285666138595 for m
ore information.
E..Exception: An error occurred (ModelError) when calling the InvokeEndpoint operation: Received server error (500) from primar
y and could not load the entire response body. See https://us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#lo
gEventViewer:group=/aws/sagemaker/Endpoints/Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23 in account 285666138595 for
more information.
E......Exception: An error occurred (ModelError) when calling the InvokeEndpoint operation: Received server error (500) from pr
imary and could not load the entire response body. See https://us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-
1#logEventViewer:group=/aws/sagemaker/Endpoints/Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23 in account 285666138595
for more information.
E..Exception: An error occurred (ModelError) when calling the InvokeEndpoint operation: Received server error (500) from primar
y and could not load the entire response body. See https://us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#lo
gEventViewer:group=/aws/sagemaker/Endpoints/Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23 in account 285666138595 for
more information.
E..Exception: An error occurred (ModelError) when calling the InvokeEndpoint operation: Received server error (500) from primar
y and could not load the entire response body. See https://us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#lo
gEventViewer:group=/aws/sagemaker/Endpoints/Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23 in account 285666138595 for
more information.
E..Exception: An error occurred (ModelError) when calling the InvokeEndpoint operation: Received server error (500) from primar
y and could not load the entire response body. See https://us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#lo
gEventViewer:group=/aws/sagemaker/Endpoints/Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23 in account 285666138595 for
more information.
E...............................................................................................
```

Wait for the update operation to complete and verify the automatic rollback.

Please note, I added a print statement for each exception (to see what was going on) so thats why instead of just "E" it is showing E then the full exception error.

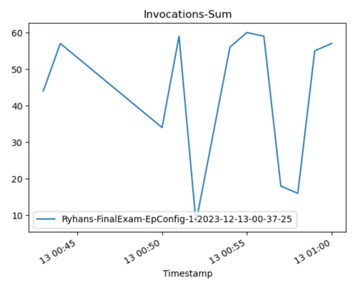
AS SEEN ABOVE, it starts with ….. Then goes to EEEE, finally rolls back to ……

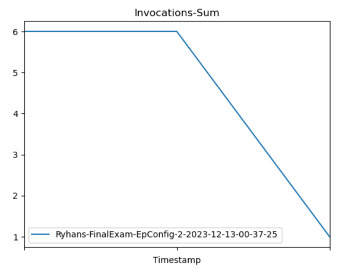
Below is the describe_endpoint output showcasing the successful ROLLBACK alarm

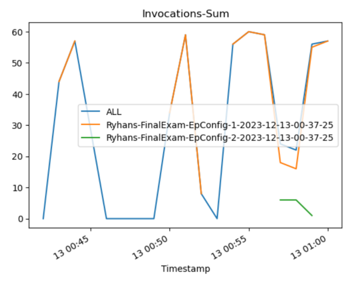


```
In [59]: wait_for_endpoint_in_service(endpoint_name)

         sm.describe_endpoint(EndpointName=endpoint_name)

         Waiting for endpoint in service

         Done!

Out[59]: {'EndpointName': 'Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23',
          'EndpointArn': 'arn:aws:sagemaker:us-east-1:285666138595:endpoint/ryhans-deployment-guardrails-canary-2023-12-13-00-40-23',
          'EndpointConfigName': 'Ryhans-FinalExam-EpConfig-1-2023-12-13-00-37-25',
          'ProductionVariants': [{'VariantName': 'AllTraffic',
            'DeployedImages': [{'SpecifiedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboost:0.90-2-cpu-py3',
              'ResolvedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboost@sha256:0d098653ff2915993d61180da0cde0ed98
         2805093463d40f30212b8050486f18',
              'ResolutionTime': datetime.datetime(2023, 12, 13, 0, 40, 24, 649000, tzinfo=tzlocal())}],
            'CurrentWeight': 1.0,
            'DesiredWeight': 1.0,
            'CurrentInstanceCount': 3,
            'DesiredInstanceCount': 3}],
          'EndpointStatus': 'InService',
          'FailureReason': 'One or more configured alarm for automatic rollback deployment is in ALARM state: [TestAlarm-ModelLatency-Ry
         hans-Deployment-Guardrails-Canary-2023-12-13-00-40-23].',
          'CreationTime': datetime.datetime(2023, 12, 13, 0, 40, 23, 976000, tzinfo=tzlocal()),
          'LastModifiedTime': datetime.datetime(2023, 12, 13, 0, 58, 41, 165000, tzinfo=tzlocal()),
          'LastDeploymentConfig': {'BlueGreenUpdatePolicy': {'TrafficRoutingConfiguration': {'Type': 'CANARY',
             'WaitIntervalInSeconds': 120,
             'CanarySize': {'Type': 'INSTANCE_COUNT', 'Value': 1}},
            'TerminationWaitInSeconds': 120,
            'MaximumExecutionTimeoutInSeconds': 1800},
           'AutoRollbackConfiguration': {'Alarms': [{'AlarmName': 'TestAlarm-5XXErrors-Ryhans-Deployment-Guardrails-Canary-2023-12-13-00
         -40-23'},
             {'AlarmName': 'TestAlarm-ModelLatency-Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23'}]}},
          'ResponseMetadata': {'RequestId': '4778639a-c46f-4921-9c8a-39a4666db708',
           'HTTPStatusCode': 200,
           'HTTPHeaders': {'x-amzn-requestid': '4778639a-c46f-4921-9c8a-39a4666db708',
            'content-type': 'application/x-amz-json-1.1',
            'content-length': '1468',
            'date': 'Wed, 13 Dec 2023 01:01:07 GMT'},
           'RetryAttempts': 0}}
```

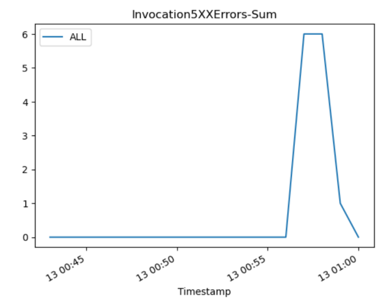
- **Invocation metrics for failure+rollback case**
- **Model 1 (ep_config-1) vs model 2 (epConfig-2) vs all**

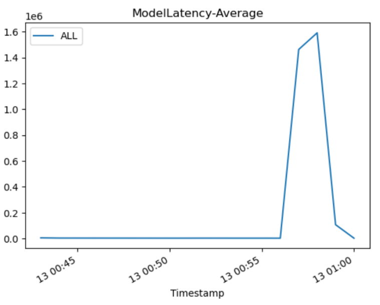






- **Below we also see 5xx errors and latency average rising due to faulty model then falling after rollback**

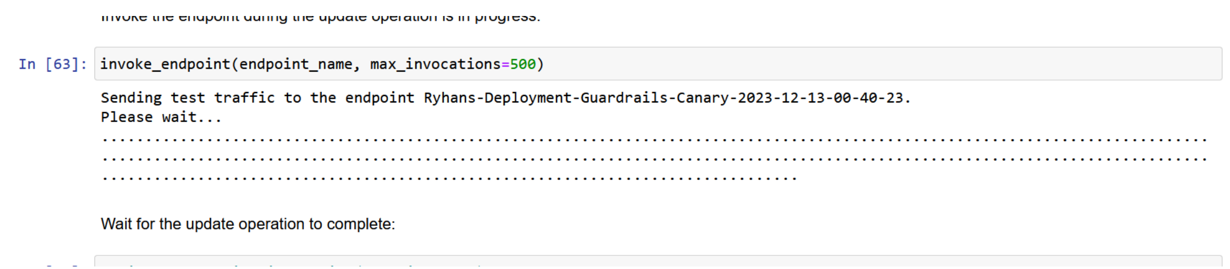




*****The second guardrail is a successful deployment, and you will show that theguardrail allows it to be deployed successfully (5 marks)***

# GUARDRAIL SUCCESS CASE

- **Below we see output for success case is ……. (no errors) after updating to ep-onfig-3 (best HP tuned model or model 2 with correct version)**

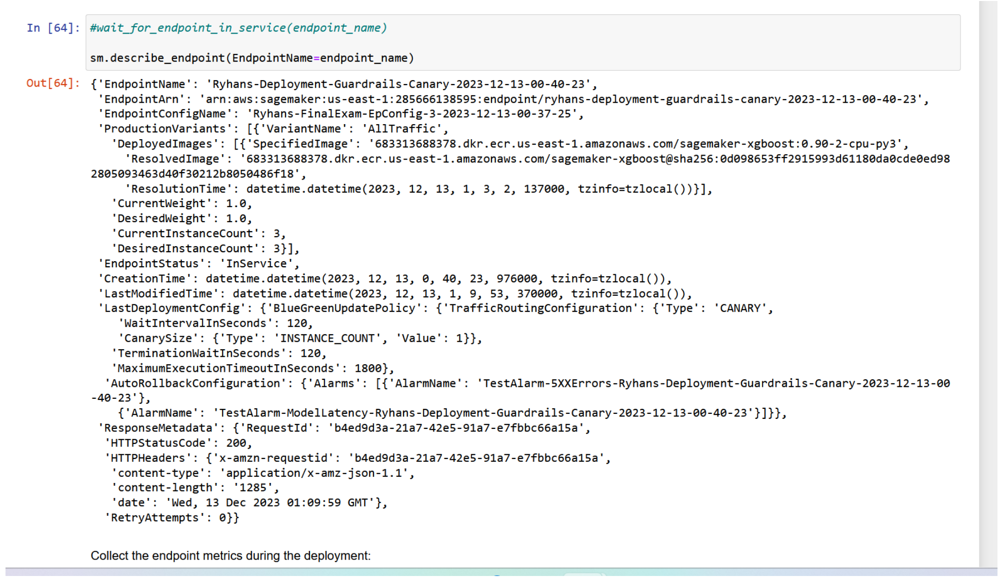


Invoke the endpoint during the update operation is in progress.

```
In [63]: invoke_endpoint(endpoint_name, max_invocations=500)

         Sending test traffic to the endpoint Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23.
         Please wait...
         ...............................................................................................
         ...............................................................................................
         ............................................................
```

Wait for the update operation to complete:

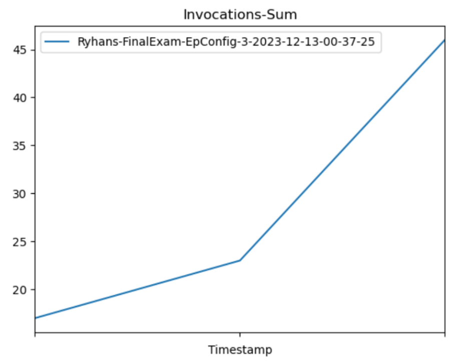
- **Output showing successful transition to ep-config-3**

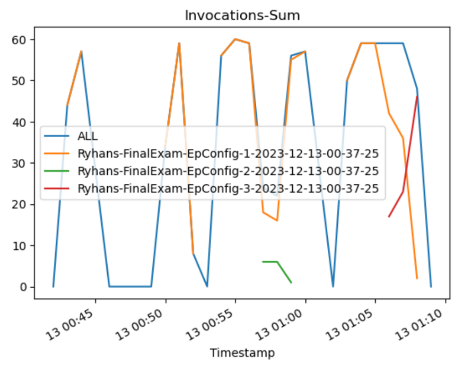


```
In [64]: #wait_for_endpoint_in_service(endpoint_name)

         sm.describe_endpoint(EndpointName=endpoint_name)

Out[64]: {'EndpointName': 'Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23',
          'EndpointArn': 'arn:aws:sagemaker:us-east-1:285666138595:endpoint/ryhans-deployment-guardrails-canary-2023-12-13-00-40-23',
          'EndpointConfigName': 'Ryhans-FinalExam-EpConfig-3-2023-12-13-00-37-25',
          'ProductionVariants': [{'VariantName': 'AllTraffic',
            'DeployedImages': [{'SpecifiedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboost:0.90-2-cpu-py3',
              'ResolvedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboost@sha256:0d098653ff2915993d61180da0cde0ed98
         2805093463d40f30212b8050486f18',
              'ResolutionTime': datetime.datetime(2023, 12, 13, 1, 3, 2, 137000, tzinfo=tzlocal())}],
            'CurrentWeight': 1.0,
            'DesiredWeight': 1.0,
            'CurrentInstanceCount': 3,
            'DesiredInstanceCount': 3}],
          'EndpointStatus': 'InService',
          'CreationTime': datetime.datetime(2023, 12, 13, 0, 40, 23, 976000, tzinfo=tzlocal()),
          'LastModifiedTime': datetime.datetime(2023, 12, 13, 1, 9, 53, 370000, tzinfo=tzlocal()),
          'LastDeploymentConfig': {'BlueGreenUpdatePolicy': {'TrafficRoutingConfiguration': {'Type': 'CANARY',
             'WaitIntervalInSeconds': 120,
             'CanarySize': {'Type': 'INSTANCE_COUNT', 'Value': 1}},
            'TerminationWaitInSeconds': 120,
            'MaximumExecutionTimeoutInSeconds': 1800},
           'AutoRollbackConfiguration': {'Alarms': [{'AlarmName': 'TestAlarm-5XXErrors-Ryhans-Deployment-Guardrails-Canary-2023-12-13-00
         -40-23'},
             {'AlarmName': 'TestAlarm-ModelLatency-Ryhans-Deployment-Guardrails-Canary-2023-12-13-00-40-23'}]}},
          'ResponseMetadata': {'RequestId': 'b4ed9d3a-21a7-42e5-91a7-e7fbbc66a15a',
           'HTTPStatusCode': 200,
           'HTTPHeaders': {'x-amzn-requestid': 'b4ed9d3a-21a7-42e5-91a7-e7fbbc66a15a',
            'content-type': 'application/x-amz-json-1.1',
            'content-length': '1285',
            'date': 'Wed, 13 Dec 2023 01:09:59 GMT'},
           'RetryAttempts': 0}}
```

Collect the endpoint metrics during the deployment:

- **Invocation Metrics**





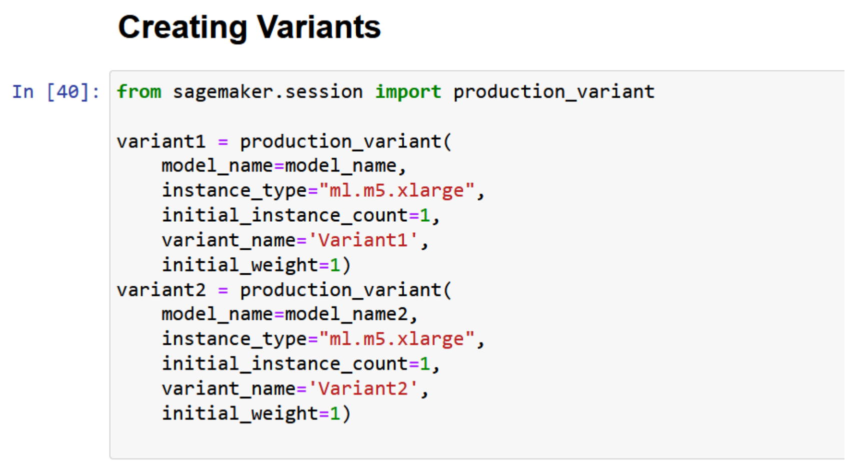
-

2. Shadow Test

***** I showed you how to use the Shadow test in the console. You can dothe same through code. Here is the link: https://sagemaker-examples.readthedocs.io/en/latest/sagemaker-shadow-variant/Shadow_variant.html (5 marks)***

• You will use the same models you have created in the guardrail. Youwill design the test in a way that the shadow variant looks better thanproduction variant and you allow that to replace the productionvariant. (4 marks)

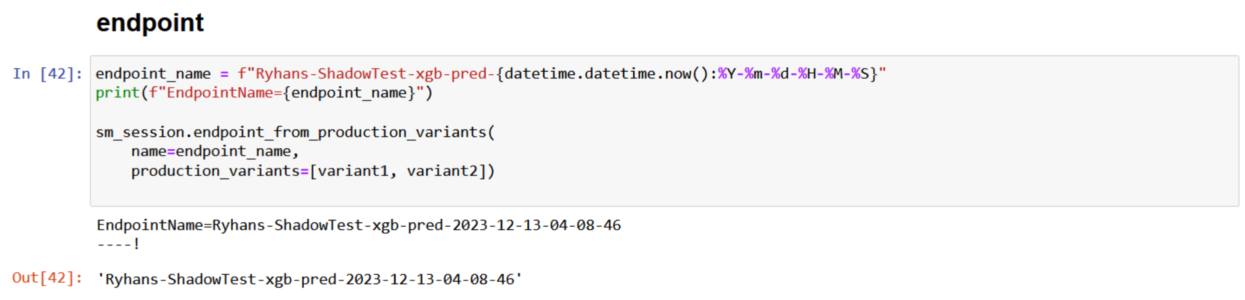
# Shadow Test using code



Creating Variants

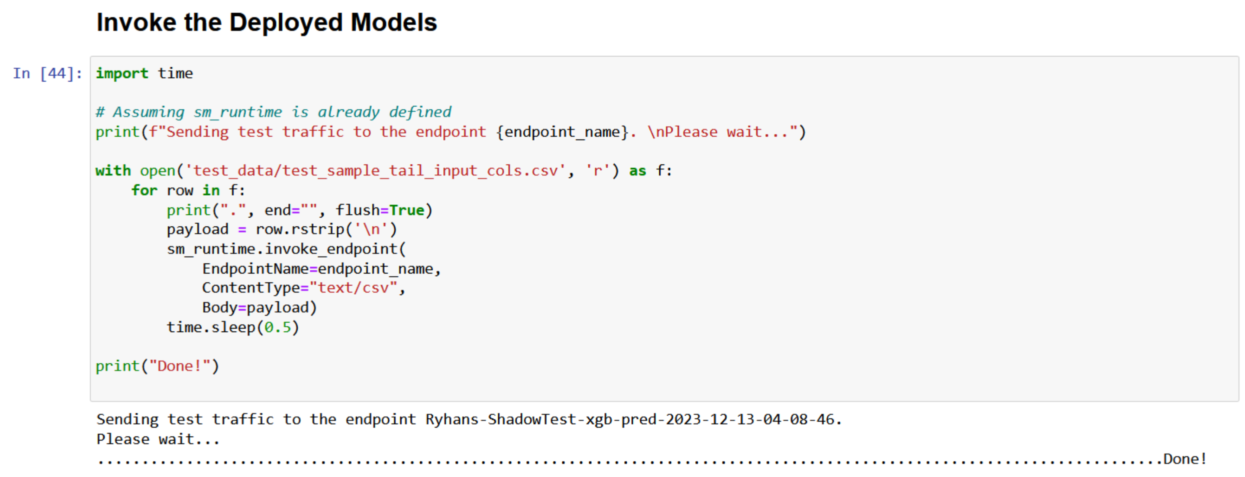
```
In [40]: from sagemaker.session import production_variant

         variant1 = production_variant(
             model_name=model_name,
             instance_type="ml.m5.xlarge",
             initial_instance_count=1,
             variant_name='Variant1',
             initial_weight=1)
         variant2 = production_variant(
             model_name=model_name2,
             instance_type="ml.m5.xlarge",
             initial_instance_count=1,
             variant_name='Variant2',
             initial_weight=1)
```

- **Invoking endpoint with both variants to compare which model is better, so we can make a decision to promote the shadow or not**



endpoint

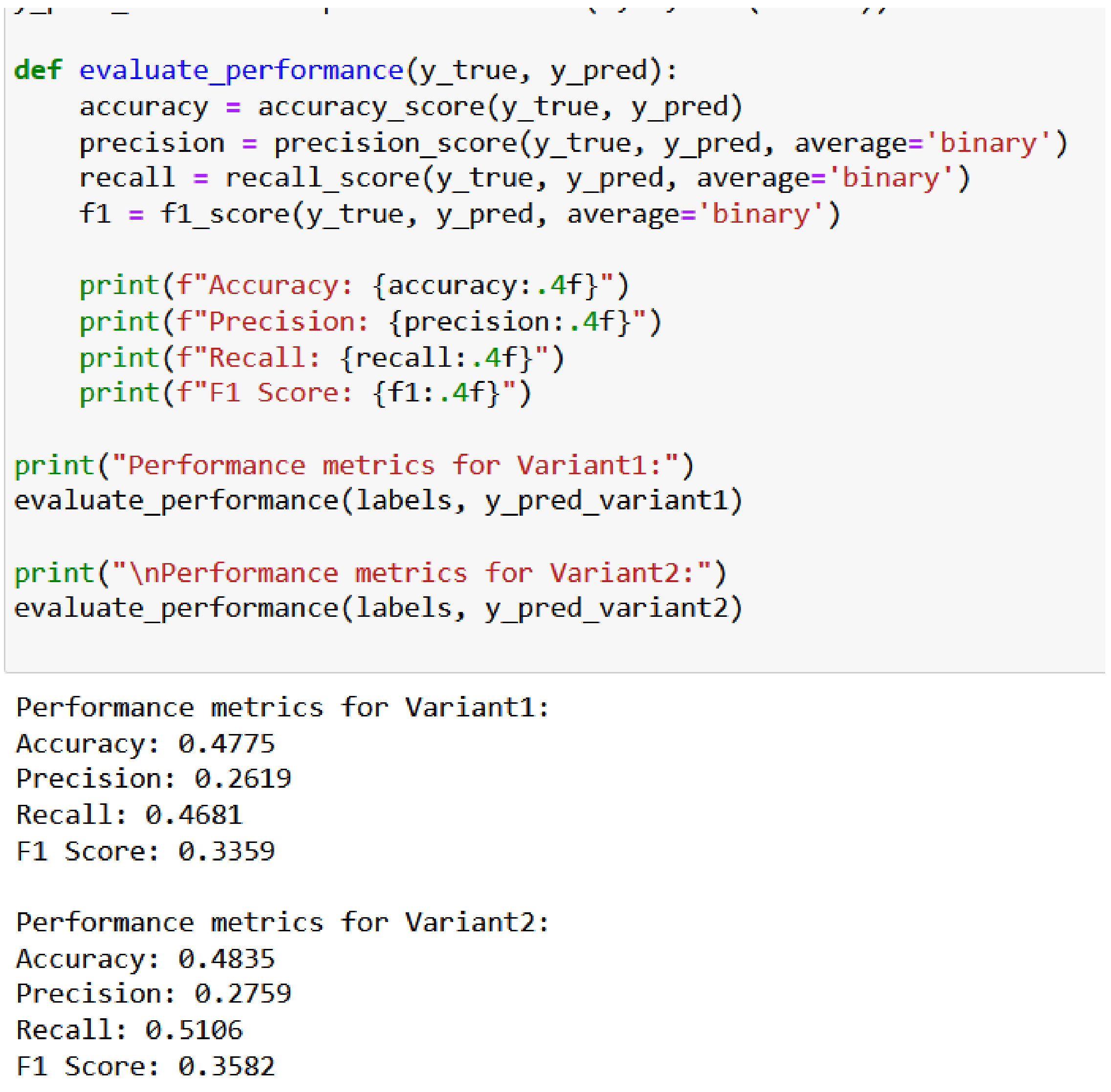
```
In [42]: endpoint_name = f"Ryhans-ShadowTest-xgb-pred-{datetime.datetime.now():%Y-%m-%d-%H-%M-%S}"
         print(f"EndpointName={endpoint_name}")

         sm_session.endpoint_from_production_variants(
             name=endpoint_name,
             production_variants=[variant1, variant2])

         EndpointName=Ryhans-ShadowTest-xgb-pred-2023-12-13-04-08-46
         ----!

Out[42]: 'Ryhans-ShadowTest-xgb-pred-2023-12-13-04-08-46'
```



Invoke the Deployed Models

```
In [44]: import time

         # Assuming sm_runtime is already defined
         print(f"Sending test traffic to the endpoint {endpoint_name}. \nPlease wait...")

         with open('test_data/test_sample_tail_input_cols.csv', 'r') as f:
             for row in f:
                 print(".", end="", flush=True)
                 payload = row.rstrip('\n')
                 sm_runtime.invoke_endpoint(
                     EndpointName=endpoint_name,
                     ContentType="text/csv",
                     Body=payload)
                 time.sleep(0.5)

         print("Done!")

         Sending test traffic to the endpoint Ryhans-ShadowTest-xgb-pred-2023-12-13-04-08-46.
         Please wait...
         ........................................................................................................................Done!
```

- **Comparing the performance of each variant**



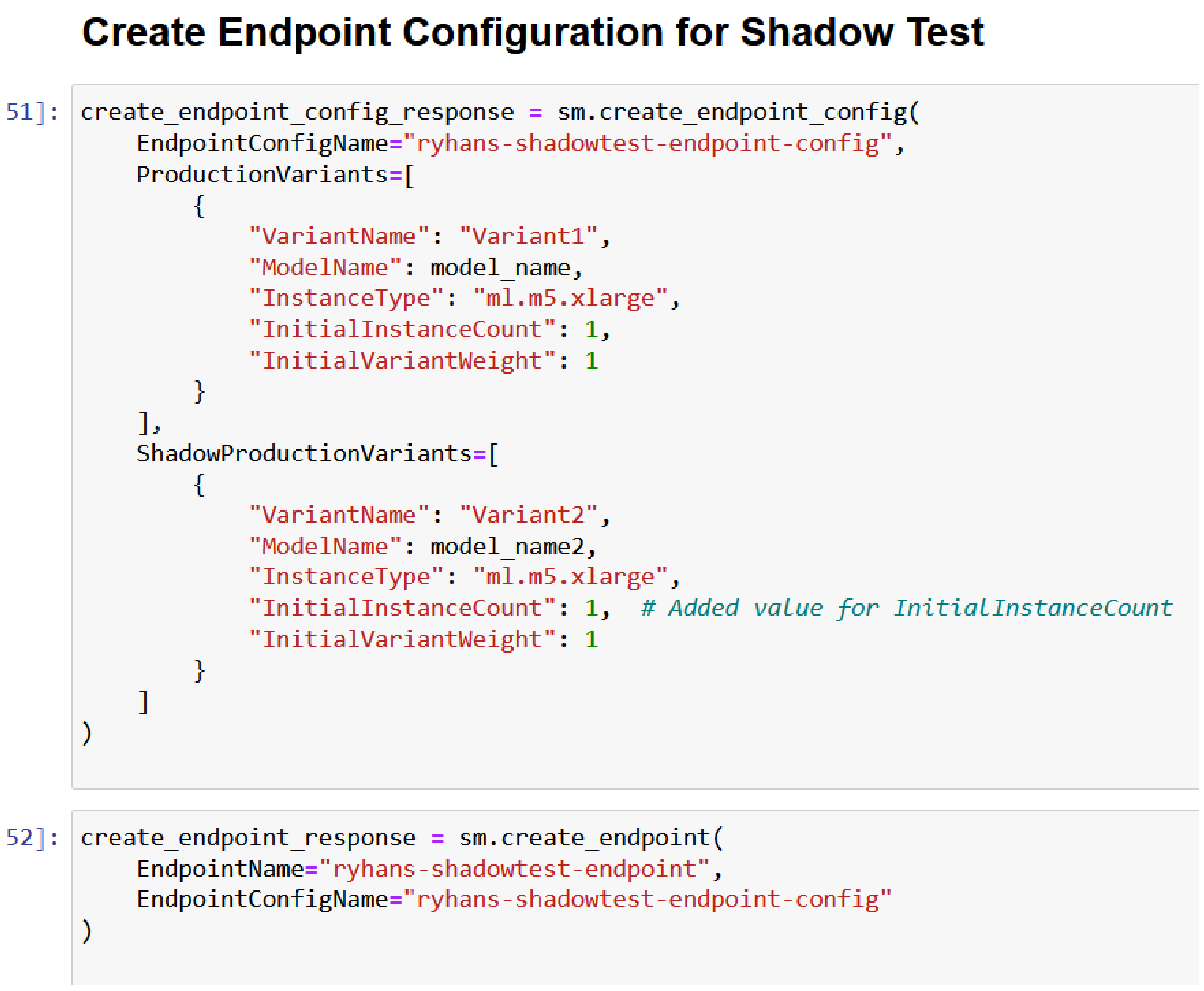
```python
def evaluate_performance(y_true, y_pred):
    accuracy = accuracy_score(y_true, y_pred)
    precision = precision_score(y_true, y_pred, average='binary')
    recall = recall_score(y_true, y_pred, average='binary')
    f1 = f1_score(y_true, y_pred, average='binary')

    print(f"Accuracy: {accuracy:.4f}")
    print(f"Precision: {precision:.4f}")
    print(f"Recall: {recall:.4f}")
    print(f"F1 Score: {f1:.4f}")

print("Performance metrics for Variant1:")
evaluate_performance(labels, y_pred_variant1)

print("\nPerformance metrics for Variant2:")
evaluate_performance(labels, y_pred_variant2)
```

```
Performance metrics for Variant1:
Accuracy: 0.4775
Precision: 0.2619
Recall: 0.4681
F1 Score: 0.3359

Performance metrics for Variant2:
Accuracy: 0.4835
Precision: 0.2759
Recall: 0.5106
F1 Score: 0.3582
```

Variant 2 seems better

so this is the one that we want to use as shadow variant.

- **Decision has been made to promote Shadow (model 2)**
- **Preparing for promotion**

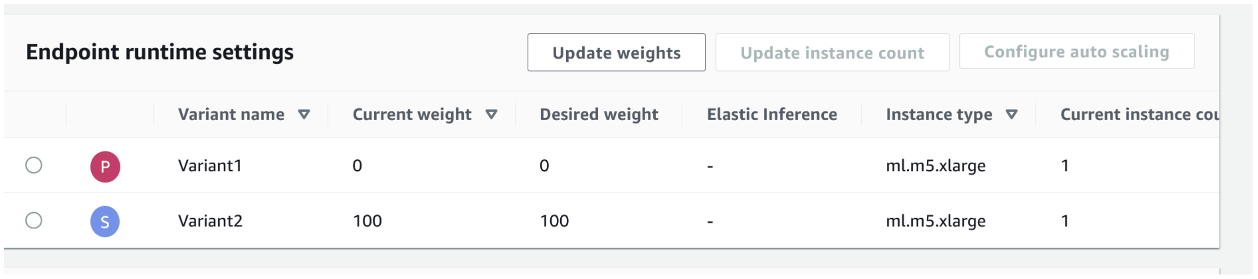


Create Endpoint Configuration for Shadow Test

```
51]: create_endpoint_config_response = sm.create_endpoint_config(
         EndpointConfigName="ryhans-shadowtest-endpoint-config",
         ProductionVariants=[
             {
                 "VariantName": "Variant1",
                 "ModelName": model_name,
                 "InstanceType": "ml.m5.xlarge",
                 "InitialInstanceCount": 1,
                 "InitialVariantWeight": 1
             }
         ],
         ShadowProductionVariants=[
             {
                 "VariantName": "Variant2",
                 "ModelName": model_name2,
                 "InstanceType": "ml.m5.xlarge",
                 "InitialInstanceCount": 1,  # Added value for InitialInstanceCount
                 "InitialVariantWeight": 1
             }
         ]
     )
```

```
52]: create_endpoint_response = sm.create_endpoint(
         EndpointName="ryhans-shadowtest-endpoint",
         EndpointConfigName="ryhans-shadowtest-endpoint-config"
     )
```

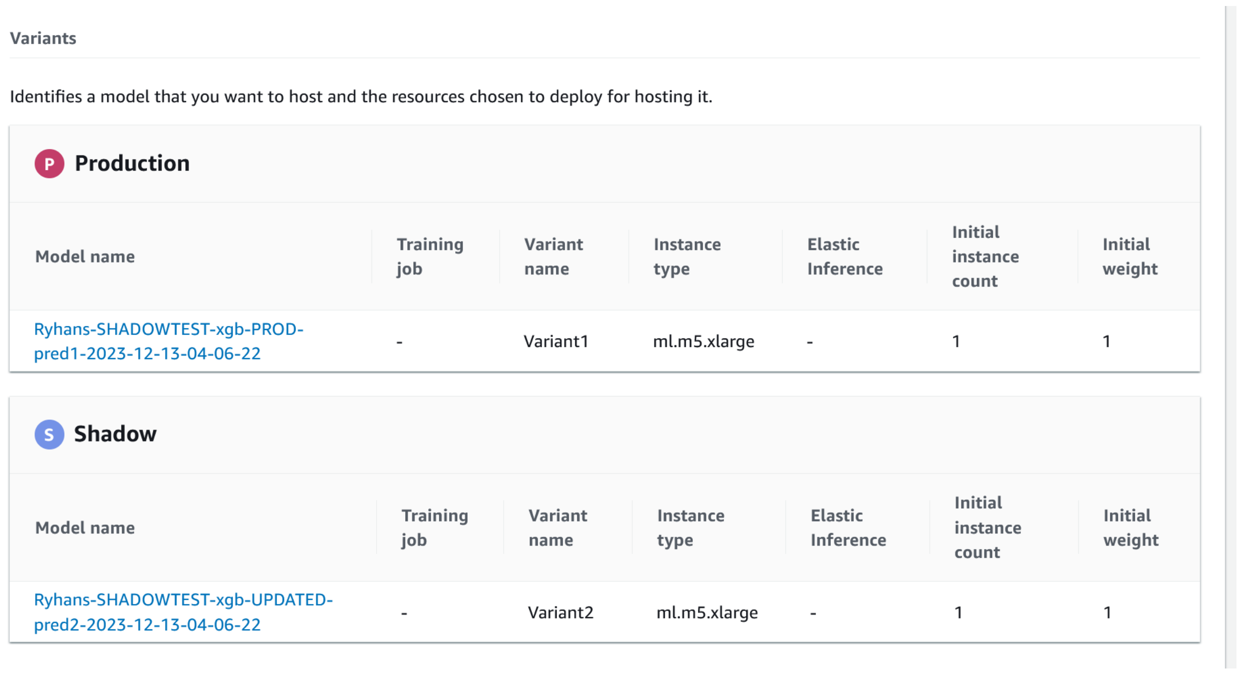
- **<u>PROMOTION SUCCESSFUL</u>**



Endpoint runtime settings

		Variant name	Current weight	Desired weight	Elastic Inference	Instance type	Current instance cou
○	P	Variant1	0	0	-	ml.m5.xlarge	1
○	S	Variant2	100	100	-	ml.m5.xlarge	1

-


Variants

Identifies a model that you want to host and the resources chosen to deploy for hosting it.

Production

Model name	Training job	Variant name	Instance type	Elastic Inference	Initial instance count	Initial weight
Ryhans-SHADOWTEST-xgb-PROD-pred1-2023-12-13-04-06-22	-	Variant1	ml.m5.xlarge	-	1	1

Shadow

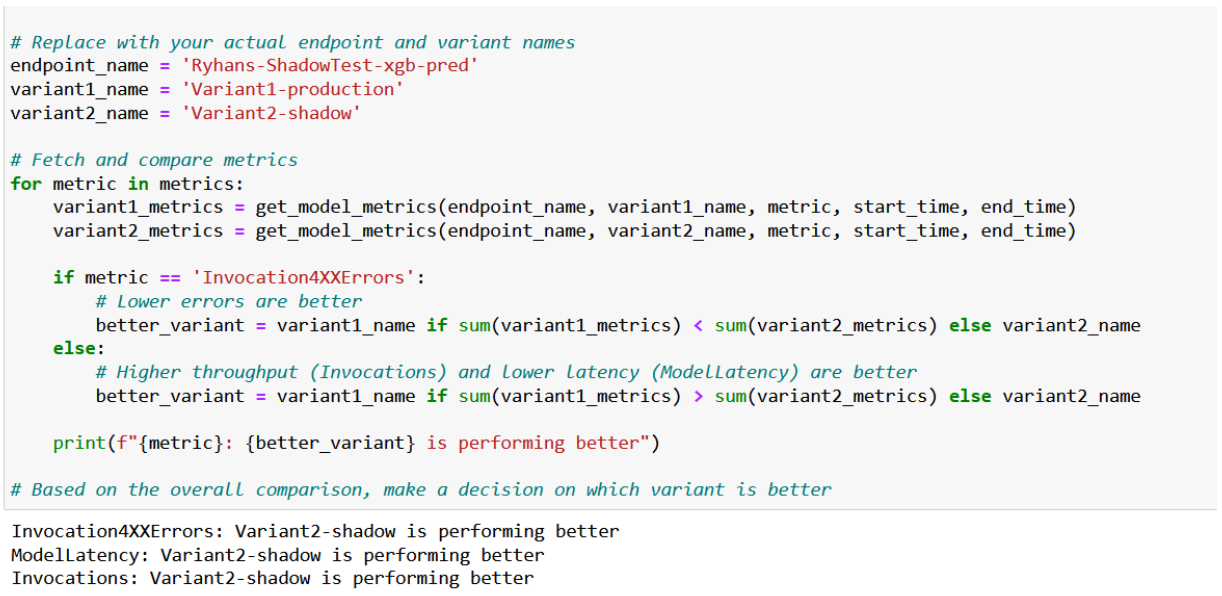
Model name	Training job	Variant name	Instance type	Elastic Inference	Initial instance count	Initial weight
Ryhans-SHADOWTEST-xgb-UPDATED-pred2-2023-12-13-04-06-22	-	Variant2	ml.m5.xlarge	-	1	1

-

# Showing that Shadow model is performing better

Is shadow variant better?

```
In [61]: import boto3
         import pandas as pd
         from datetime import datetime, timedelta

         # Initialize the CloudWatch client
         cloudwatch = boto3.client('cloudwatch')

         def get_model_metrics(endpoint_name, variant_name, metric_name, start_time, end_time):
             """ Fetch metrics for a specific model variant from CloudWatch """
             response = cloudwatch.get_metric_data(
                 MetricDataQueries=[
                     {
                         'Id': 'm1',
                         'MetricStat': {
                             'Metric': {
                                 'Namespace': 'AWS/SageMaker',
                                 'MetricName': metric_name,
                                 'Dimensions': [
                                     {'Name': 'EndpointName', 'Value': endpoint_name},
                                     {'Name': 'VariantName', 'Value': variant_name}
                                 ]
                             },
                             'Period': 300,  # Period in seconds
                             'Stat': 'Average',
                         },
                         'ReturnData': True,
                     },
                 ],
                 StartTime=start_time,
                 EndTime=end_time
             )
             return response['MetricDataResults'][0]['Values']

         # Define the metrics to be fetched
         metrics = ['Invocation4XXErrors', 'ModelLatency', 'Invocations']

         # Set time range for metrics
         end_time = datetime.utcnow()
         start_time = end_time - timedelta(days=1)
```



```python
# Replace with your actual endpoint and variant names
endpoint_name = 'Ryhans-ShadowTest-xgb-pred'
variant1_name = 'Variant1-production'
variant2_name = 'Variant2-shadow'

# Fetch and compare metrics
for metric in metrics:
    variant1_metrics = get_model_metrics(endpoint_name, variant1_name, metric, start_time, end_time)
    variant2_metrics = get_model_metrics(endpoint_name, variant2_name, metric, start_time, end_time)

    if metric == 'Invocation4XXErrors':
        # Lower errors are better
        better_variant = variant1_name if sum(variant1_metrics) < sum(variant2_metrics) else variant2_name
    else:
        # Higher throughput (Invocations) and lower latency (ModelLatency) are better
        better_variant = variant1_name if sum(variant1_metrics) > sum(variant2_metrics) else variant2_name

    print(f"{metric}: {better_variant} is performing better")

# Based on the overall comparison, make a decision on which variant is better
```

```
Invocation4XXErrors: Variant2-shadow is performing better
ModelLatency: Variant2-shadow is performing better
Invocations: Variant2-shadow is performing better
```

-

- **AS SEEN IN THE OUTPUT: shadow variant is better than production variant in terms of invocation and latency metrics.**
- **<u>Shadow Test Completed with Success.</u>**

Thank you