In [1]:
```
!pip install -U awscli
!pip install sagemaker
```

```
    ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 11.8/11.8 MB 89.8 MB/s eta 0:00:00:00:0100:01
Installing collected packages: botocore, awscli
  Attempting uninstall: botocore
    Found existing installation: botocore 1.33.4
    Uninstalling botocore-1.33.4:
      Successfully uninstalled botocore-1.33.4
  Attempting uninstall: awscli
    Found existing installation: awscli 1.31.4
    Uninstalling awscli-1.31.4:
      Successfully uninstalled awscli-1.31.4
Successfully installed awscli-1.31.11 botocore-1.33.11
Requirement already satisfied: sagemaker in /home/ec2-user/anaconda3/envs/python3/lib/python3.10/site
-packages (2.199.0)
Requirement already satisfied: attrs<24,>=23.1.0 in /home/ec2-user/anaconda3/envs/python3/lib/python
3.10/site-packages (from sagemaker) (23.1.0)
Requirement already satisfied: boto3<2.0,>=1.33.3 in /home/ec2-user/anaconda3/envs/python3/lib/python
3.10/site-packages (from sagemaker) (1.33.4)
Requirement already satisfied: cloudpickle==2.2.1 in /home/ec2-user/anaconda3/envs/python3/lib/python
3.10/site-packages (from sagemaker) (2.2.1)
Requirement already satisfied: google-pasta in /home/ec2-user/anaconda3/envs/python3/lib/python3.10/s
```

In [7]:
```
!aws s3 cp s3://day09-project1/sagemaker/Employess-Promotion/output/xgboost-tuningjob-26-21-56-00-010-62a
!aws s3 cp s3://day09-project1/sagemaker/Employess-Promotion/output/xgboost-tuningjob-26-21-56-00-001-65b

!aws s3 cp s3://day09-project1/sagemaker/Employess-Promotion/train/train.csv train_data/
!aws s3 cp s3://day09-project1/sagemaker/Employess-Promotion/test/test.csv test_data/
```

```
download: s3://day09-project1/sagemaker/Employess-Promotion/output/xgboost-tuningjob-26-21-56-00-010-62a
4af05/output/model.tar.gz to model/model1/model.tar.gz
download: s3://day09-project1/sagemaker/Employess-Promotion/output/xgboost-tuningjob-26-21-56-00-001-65b
129a3/output/model.tar.gz to model/model2/model.tar.gz
download: s3://day09-project1/sagemaker/Employess-Promotion/train/train.csv to train_data/train.csv
download: s3://day09-project1/sagemaker/Employess-Promotion/test/test.csv to test_data/test.csv
```

In [1]:
```
%matplotlib inline

import time
import os
import boto3
import botocore
import re
import json
from datetime import datetime, timedelta, timezone
from sagemaker import get_execution_role, session
from sagemaker.s3 import S3Downloader, S3Uploader

region = boto3.Session().region_name

# You can use a different IAM role with "SageMakerFullAccess" policy for this notebook
role = get_execution_role()
print(f"Execution role: {role}")

sm_session = session.Session(boto3.Session())
sm = boto3.Session().client("sagemaker")
sm_runtime = boto3.Session().client("sagemaker-runtime")

# You can use a different bucket, but make sure the role you chose for this notebook
# has the s3:PutObject permissions. This is the bucket into which the model artifacts will be uploaded
# bucket = sm_session.default_bucket()
# buckegt
# print(bucket)
# prefix = "sagemaker/Final-Project-Deployment-Guardrails-Canary"
```

```
Matplotlib is building the font cache; this may take a moment.
/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages/pandas/core/computation/expressions.p
y:21: UserWarning: Pandas requires version '2.8.0' or newer of 'numexpr' (version '2.7.3' currently inst
alled).
  from pandas.core.computation.check import NUMEXPR_INSTALLED

sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemaker/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /home/ec2-user/.config/sagemaker/confi
g.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemaker/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /home/ec2-user/.config/sagemaker/confi
g.yaml
Execution role: arn:aws:iam::585522057818:role/fast-ai-academic-11-Student-Azure
sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemaker/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /home/ec2-user/.config/sagemaker/confi
g.yaml
```

In [2]:
```
# bucket = "day09-project1"
# print(bucket)
# prefix = "sagemaker/Employess-Promotion/output"
bucket = sm_session.default_bucket()
print(bucket)
prefix = "sagemaker/Final-Project-Deployment-Guardrails-Canary"
```

```
sagemaker-us-east-1-585522057818
```

In [3]:
```
model_url = S3Uploader.upload(
    local_path="model/model1/model.tar.gz",    # best model
    desired_s3_uri=f"s3://{bucket}/{prefix}/model1",
)
model_url2 = S3Uploader.upload(
    local_path="model/model2/model.tar.gz",
    desired_s3_uri=f"s3://{bucket}/{prefix}model2",
)

print(f"Model URI 1: {model_url}")
print(f"Model URI 2: {model_url2}")
```

```
sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemaker/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /home/ec2-user/.config/sagemaker/confi
g.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemaker/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /home/ec2-user/.config/sagemaker/confi
g.yaml
Model URI 1: s3://sagemaker-us-east-1-585522057818/sagemaker/Final-Project-Deployment-Guardrails-Canary/
model1/model.tar.gz
Model URI 2: s3://sagemaker-us-east-1-585522057818/sagemaker/Final-Project-Deployment-Guardrails-Canarym
odel2/model.tar.gz
```

In [5]:
```
from sagemaker import image_uris

image_uri = image_uris.retrieve("xgboost", boto3.Session().region_name, "1.5-1")

# using newer version of XGBoost which is incompatible, in order to simulate model faults
image_uri2 = image_uris.retrieve("image-classification-neo", boto3.Session().region_name)
image_uri3 = image_uris.retrieve("xgboost", boto3.Session().region_name, "1.3-1")

print(f"Model Image 1: {image_uri}")
print(f"Model Image 2: {image_uri2}")
print(f"Model Image 3: {image_uri3}")
```

```
Model Image 1: 683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboost:1.5-1
Model Image 2: 785573368785.dkr.ecr.us-east-1.amazonaws.com/image-classification-neo:latest
Model Image 3: 683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboost:1.3-1
```

In [6]:
```
model_name = f"Final-xgb-emp-pred-{datetime.now():%Y-%m-%d-%H-%M-%S}"
model_name2 = f"Final-xgb-emp-pred2-{datetime.now():%Y-%m-%d-%H-%M-%S}"
model_name3 = f"Final-xgb-emp-pred3-{datetime.now():%Y-%m-%d-%H-%M-%S}"

print(f"Model Name 1: {model_name}")
print(f"Model Name 2: {model_name2}")
print(f"Model Name 3: {model_name3}")

resp = sm.create_model(
    ModelName=model_name,
    ExecutionRoleArn=role,
    Containers=[{"Image": image_uri, "ModelDataUrl": model_url}],
)
print(f"Created Model: {resp}")

resp = sm.create_model(
    ModelName=model_name2,
    ExecutionRoleArn=role,
    Containers=[{"Image": image_uri2, "ModelDataUrl": model_url2}],
)
print(f"Created Model: {resp}")

resp = sm.create_model(
    ModelName=model_name3,
    ExecutionRoleArn=role,
    Containers=[{"Image": image_uri3, "ModelDataUrl": model_url2}],
)
print(f"Created Model: {resp}")
```

```
Model Name 1: Final-xgb-emp-pred-2023-12-13-20-15-21
Model Name 2: Final-xgb-emp-pred2-2023-12-13-20-15-21
Model Name 3: Final-xgb-emp-pred3-2023-12-13-20-15-21
Created Model: {'ModelArn': 'arn:aws:sagemaker:us-east-1:585522057818:model/final-xgb-emp-pred-2023-12-1
3-20-15-21', 'ResponseMetadata': {'RequestId': '73e6fa71-b983-4a3a-8523-facc3bd3524e', 'HTTPStatusCode':
200, 'HTTPHeaders': {'x-amzn-requestid': '73e6fa71-b983-4a3a-8523-facc3bd3524e', 'content-type': 'applic
ation/x-amz-json-1.1', 'content-length': '100', 'date': 'Wed, 13 Dec 2023 20:15:21 GMT'}, 'RetryAttempt
s': 0}}
Created Model: {'ModelArn': 'arn:aws:sagemaker:us-east-1:585522057818:model/final-xgb-emp-pred2-2023-12-
13-20-15-21', 'ResponseMetadata': {'RequestId': 'bc802a5b-2852-44ba-816a-c0e9cbb0f00a', 'HTTPStatusCod
e': 200, 'HTTPHeaders': {'x-amzn-requestid': 'bc802a5b-2852-44ba-816a-c0e9cbb0f00a', 'content-type': 'ap
plication/x-amz-json-1.1', 'content-length': '101', 'date': 'Wed, 13 Dec 2023 20:15:23 GMT'}, 'RetryAtte
mpts': 1}}
Created Model: {'ModelArn': 'arn:aws:sagemaker:us-east-1:585522057818:model/final-xgb-emp-pred3-2023-12-
13-20-15-21', 'ResponseMetadata': {'RequestId': '4734e97d-8f80-4103-9608-b4ba7c7569c5', 'HTTPStatusCod
e': 200, 'HTTPHeaders': {'x-amzn-requestid': '4734e97d-8f80-4103-9608-b4ba7c7569c5', 'content-type': 'ap
plication/x-amz-json-1.1', 'content-length': '101', 'date': 'Wed, 13 Dec 2023 20:15:24 GMT'}, 'RetryAtte
mpts': 1}}
```

Endpoint Config

In [7]:
```
ep_config_name = f"Final-EpConfig-1-{datetime.now():%Y-%m-%d-%H-%M-%S}"
ep_config_name2 = f"Final-EpConfig-2-{datetime.now():%Y-%m-%d-%H-%M-%S}"
ep_config_name3 = f"Final-EpConfig-3-{datetime.now():%Y-%m-%d-%H-%M-%S}"

print(f"Endpoint Config 1: {ep_config_name}")
print(f"Endpoint Config 2: {ep_config_name2}")
print(f"Endpoint Config 3: {ep_config_name3}")

resp = sm.create_endpoint_config(
    EndpointConfigName=ep_config_name,
    ProductionVariants=[
        {
            "VariantName": "AllTraffic",
            "ModelName": model_name,
            "InstanceType": "ml.m5.xlarge",
            "InitialInstanceCount": 3,
        }
    ],
)
print(f"Created Endpoint Config: {resp}")
time.sleep(5)

resp = sm.create_endpoint_config(
    EndpointConfigName=ep_config_name2,
    ProductionVariants=[
        {
            "VariantName": "AllTraffic",
            "ModelName": model_name2,
            "InstanceType": "ml.m5.xlarge",
            "InitialInstanceCount": 3,
        }
    ],
)
print(f"Created Endpoint Config: {resp}")
time.sleep(5)

resp = sm.create_endpoint_config(
    EndpointConfigName=ep_config_name3,
    ProductionVariants=[
        {
            "VariantName": "AllTraffic",
            "ModelName": model_name3,
            "InstanceType": "ml.m5.xlarge",
            "InitialInstanceCount": 3,
        }
    ],
)
print(f"Created Endpoint Config: {resp}")
time.sleep(5)
```

```
Endpoint Config 1: Final-EpConfig-1-2023-12-13-20-16-27
Endpoint Config 2: Final-EpConfig-2-2023-12-13-20-16-27
Endpoint Config 3: Final-EpConfig-3-2023-12-13-20-16-27
Created Endpoint Config: {'EndpointConfigArn': 'arn:aws:sagemaker:us-east-1:585522057818:endpoint-confi
g/final-epconfig-1-2023-12-13-20-16-27', 'ResponseMetadata': {'RequestId': '6aed3423-c48e-4184-9acb-b76f
ad6f2a4b', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': '6aed3423-c48e-4184-9acb-b76fad6f2
a4b', 'content-type': 'application/x-amz-json-1.1', 'content-length': '117', 'date': 'Wed, 13 Dec 2023 2
0:16:27 GMT'}, 'RetryAttempts': 0}}
Created Endpoint Config: {'EndpointConfigArn': 'arn:aws:sagemaker:us-east-1:585522057818:endpoint-confi
g/final-epconfig-2-2023-12-13-20-16-27', 'ResponseMetadata': {'RequestId': '1f201202-4d83-4371-b516-ef17
b6537ebb', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': '1f201202-4d83-4371-b516-ef17b6537
ebb', 'content-type': 'application/x-amz-json-1.1', 'content-length': '117', 'date': 'Wed, 13 Dec 2023 2
0:16:32 GMT'}, 'RetryAttempts': 0}}
Created Endpoint Config: {'EndpointConfigArn': 'arn:aws:sagemaker:us-east-1:585522057818:endpoint-confi
g/final-epconfig-3-2023-12-13-20-16-27', 'ResponseMetadata': {'RequestId': '0eb10d6a-a892-45b6-89a8-0b59
fdd04223', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': '0eb10d6a-a892-45b6-89a8-0b59fdd04
223', 'content-type': 'application/x-amz-json-1.1', 'content-length': '117', 'date': 'Wed, 13 Dec 2023 2
0:16:38 GMT'}, 'RetryAttempts': 0}}
```

create endpoint

In [8]:
```
endpoint_name = f"Final-Deployment-Guardrails-Canary-{datetime.now():%Y-%m-%d-%H-%M-%S}"
print(f"Endpoint Name: {endpoint_name}")

resp = sm.create_endpoint(EndpointName=endpoint_name, EndpointConfigName=ep_config_name)
print(f"\nCreated Endpoint: {resp}")
```

```
Endpoint Name: Final-Deployment-Guardrails-Canary-2023-12-13-20-16-49

Created Endpoint: {'EndpointArn': 'arn:aws:sagemaker:us-east-1:585522057818:endpoint/final-deployment-gu
ardrails-canary-2023-12-13-20-16-49', 'ResponseMetadata': {'RequestId': '8efeb991-2123-4373-8a86-bb780c8
7d7a4', 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': '8efeb991-2123-4373-8a86-bb780c87d7a
4', 'content-type': 'application/x-amz-json-1.1', 'content-length': '122', 'date': 'Wed, 13 Dec 2023 20:
16:50 GMT'}, 'RetryAttempts': 0}}
```

In [9]:
```
def wait_for_endpoint_in_service(endpoint_name):
    print("Waiting for endpoint in service")
    while True:
        details = sm.describe_endpoint(EndpointName=endpoint_name)
        status = details["EndpointStatus"]
        if status in ["InService", "Failed"]:
            print("\nDone!")
            break
        print(".", end="", flush=True)
        time.sleep(30)


wait_for_endpoint_in_service(endpoint_name)

sm.describe_endpoint(EndpointName=endpoint_name)
```

```
Waiting for endpoint in service

Done!
```

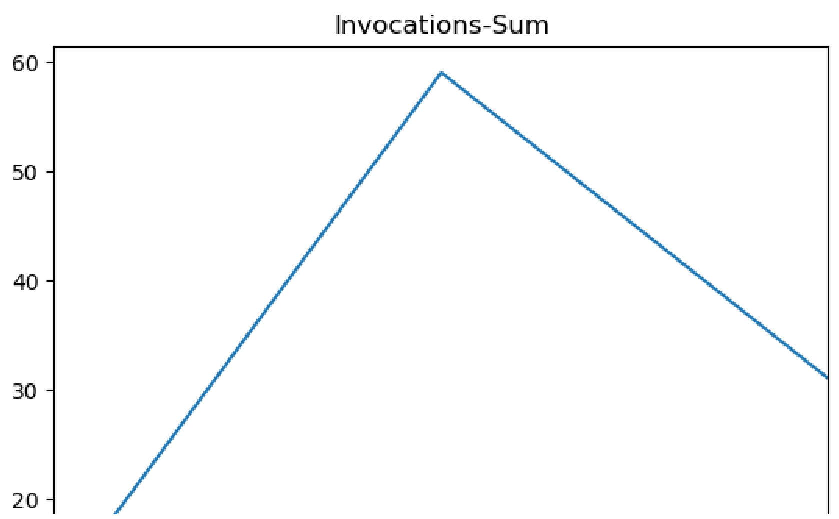
Out[9]:
```
{'EndpointName': 'Final-Deployment-Guardrails-Canary-2023-12-13-20-16-49',
 'EndpointArn': 'arn:aws:sagemaker:us-east-1:585522057818:endpoint/final-deployment-guardrails-canary-20
23-12-13-20-16-49',
 'EndpointConfigName': 'Final-EpConfig-1-2023-12-13-20-16-27',
 'ProductionVariants': [{'VariantName': 'AllTraffic',
   'DeployedImages': [{'SpecifiedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboos
t:1.5-1',
     'ResolvedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboost@sha256:43d2c4ea768
1f36634727803636d5b7d58d85bceef43aa79f78bfde851d59c19',
     'ResolutionTime': datetime.datetime(2023, 12, 13, 20, 16, 50, 860000, tzinfo=tzlocal())}],
   'CurrentWeight': 1.0,
   'DesiredWeight': 1.0,
   'CurrentInstanceCount': 3,
   'DesiredInstanceCount': 3}],
 'EndpointStatus': 'InService',
 'CreationTime': datetime.datetime(2023, 12, 13, 20, 16, 50, 13000, tzinfo=tzlocal()),
 'LastModifiedTime': datetime.datetime(2023, 12, 13, 20, 19, 36, 274000, tzinfo=tzlocal()),
 'ResponseMetadata': {'RequestId': 'c513dae5-16cf-422d-9e3a-71573ae5b3be',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': 'c513dae5-16cf-422d-9e3a-71573ae5b3be',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '788',
   'date': 'Wed, 13 Dec 2023 20:20:37 GMT'},
  'RetryAttempts': 0}}
```

In [10]:
```python
def invoke_endpoint(
    endpoint_name, max_invocations=600, wait_interval_sec=1, should_raise_exp=False
):
    print(f"Sending test traffic to the endpoint {endpoint_name}. \nPlease wait...")

    count = 0
    with open("test_data/test.csv", "r") as f:
        for row in f:
            payload = row.rstrip("\n")
            try:
                response = sm_runtime.invoke_endpoint(
                    EndpointName=endpoint_name, ContentType="text/csv", Body=payload
                )
                response["Body"].read()
                print(".", end="", flush=True)
            except Exception as e:
#                 print(e)
                print("E", end="", flush=True)
                if should_raise_exp:
                    raise e
            count += 1
            if count > max_invocations:
                break
            time.sleep(wait_interval_sec)

    print("\nDone!")


invoke_endpoint(endpoint_name, max_invocations=100)
```

```
Sending test traffic to the endpoint Final-Deployment-Guardrails-Canary-2023-12-13-20-16-49.
Please wait...
.....................................................................................................
Done!
```

In [11]:
```
import pandas as pd

cw = boto3.Session().client("cloudwatch", region_name=region)


def get_sagemaker_metrics(
    endpoint_name,
    endpoint_config_name,
    variant_name,
    metric_name,
    statistic,
    start_time,
    end_time,
):
    dimensions = [
        {"Name": "EndpointName", "Value": endpoint_name},
        {"Name": "VariantName", "Value": variant_name},
    ]
    if endpoint_config_name is not None:
        dimensions.append({"Name": "EndpointConfigName", "Value": endpoint_config_name})
    metrics = cw.get_metric_statistics(
        Namespace="AWS/SageMaker",
        MetricName=metric_name,
        StartTime=start_time,
        EndTime=end_time,
        Period=60,
        Statistics=[statistic],
        Dimensions=dimensions,
    )
    rename = endpoint_config_name if endpoint_config_name is not None else "ALL"
    if len(metrics["Datapoints"]) == 0:
        return
    return (
        pd.DataFrame(metrics["Datapoints"])
        .sort_values("Timestamp")
        .set_index("Timestamp")
        .drop(["Unit"], axis=1)
        .rename(columns={statistic: rename})
    )


def plot_endpoint_invocation_metrics(
    endpoint_name,
    endpoint_config_name,
    variant_name,
    metric_name,
    statistic,
    start_time=None,
):
    start_time = start_time or datetime.now(timezone.utc) - timedelta(minutes=60)
    end_time = datetime.now(timezone.utc)
    metrics_variants = get_sagemaker_metrics(
        endpoint_name,
        endpoint_config_name,
        variant_name,
        metric_name,
        statistic,
        start_time,
        end_time,
    )
    if metrics_variants is None:
        return
    metrics_variants.plot(title=f"{metric_name}-{statistic}")
    return metrics_variants
```

In [12]:
```
invocation_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, ep_config_name, "AllTraffic", "Invocations", "Sum"
)
invocation_4xx_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, None, "AllTraffic", "Invocation4XXErrors", "Sum"
)
invocation_5xx_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, None, "AllTraffic", "Invocation5XXErrors", "Sum"
)
model_latency_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, None, "AllTraffic", "ModelLatency", "Average"
)
overhead_latency_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, None, "AllTraffic", "OverheadLatency", "Average"
)
```



In [13]:
```
def create_auto_rollback_alarm(
    alarm_name, endpoint_name, variant_name, metric_name, statistic, threshold
):
    cw.put_metric_alarm(
        AlarmName=alarm_name,
        AlarmDescription="Test SageMaker endpoint deployment auto-rollback alarm",
        ActionsEnabled=False,
        Namespace="AWS/SageMaker",
        MetricName=metric_name,
        Statistic=statistic,
        Dimensions=[
            {"Name": "EndpointName", "Value": endpoint_name},
            {"Name": "VariantName", "Value": variant_name},
        ],
        Period=60,
        EvaluationPeriods=1,
        Threshold=threshold,
        ComparisonOperator="GreaterThanOrEqualToThreshold",
        TreatMissingData="notBreaching",
    )
```

In [14]:
```
error_alarm = f"TestAlarm-5XXErrors-{endpoint_name}"
latency_alarm = f"TestAlarm-ModelLatency-{endpoint_name}"

# alarm on 1% 5xx error rate for 1 minute
create_auto_rollback_alarm(
    error_alarm, endpoint_name, "AllTraffic", "Invocation5XXErrors", "Average", 1
)
# alarm on model latency >= 10 ms for 1 minute
create_auto_rollback_alarm(
    latency_alarm, endpoint_name, "AllTraffic", "ModelLatency", "Average", 10000
)
```

In [15]:
```
cw.describe_alarms(AlarmNames=[error_alarm, latency_alarm])
time.sleep(60)
```

Update endpoint

In [16]:
```
canary_deployment_config = {
    "BlueGreenUpdatePolicy": {
        "TrafficRoutingConfiguration": {
            "Type": "CANARY",
            "CanarySize": {
                "Type": "INSTANCE_COUNT",  # or use "CAPACITY_PERCENT" as 30%, 50%
                "Value": 1,
            },
            "WaitIntervalInSeconds": 300,  # wait for 5 minutes before enabling traffic on the rest of fl
        },
        "TerminationWaitInSeconds": 120,  # wait for 2 minutes before terminating the old stack
        "MaximumExecutionTimeoutInSeconds": 1800,  # maximum timeout for deployment
    },
    "AutoRollbackConfiguration": {
        "Alarms": [{"AlarmName": error_alarm}, {"AlarmName": latency_alarm}],
    },
}

# update endpoint request with new DeploymentConfig parameter
sm.update_endpoint(
    EndpointName=endpoint_name,
    EndpointConfigName=ep_config_name2,
    DeploymentConfig=canary_deployment_config,
)
```

Out[16]:
```
{'EndpointArn': 'arn:aws:sagemaker:us-east-1:585522057818:endpoint/final-deployment-guardrails-canary-20
23-12-13-20-16-49',
 'ResponseMetadata': {'RequestId': 'b81d8a12-5fa1-4bee-b6f1-5dcb9fcbadd7',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': 'b81d8a12-5fa1-4bee-b6f1-5dcb9fcbadd7',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '122',
   'date': 'Wed, 13 Dec 2023 20:26:57 GMT'},
  'RetryAttempts': 0}}
```

In [17]:
```
sm.describe_endpoint(EndpointName=endpoint_name)
```

```
   'DesiredWeight': 1.0,
   'CurrentInstanceCount': 3,
   'DesiredInstanceCount': 3}],
 'EndpointStatus': 'Updating',
 'CreationTime': datetime.datetime(2023, 12, 13, 20, 16, 50, 13000, tzinfo=tzlocal()),
 'LastModifiedTime': datetime.datetime(2023, 12, 13, 20, 26, 58, 391000, tzinfo=tzlocal()),
 'LastDeploymentConfig': {'BlueGreenUpdatePolicy': {'TrafficRoutingConfiguration': {'Type': 'CANARY',
    'WaitIntervalInSeconds': 300,
    'CanarySize': {'Type': 'INSTANCE_COUNT', 'Value': 1}},
   'TerminationWaitInSeconds': 120,
   'MaximumExecutionTimeoutInSeconds': 1800},
  'AutoRollbackConfiguration': {'Alarms': [{'AlarmName': 'TestAlarm-5XXErrors-Final-Deployment-Guardr
ails-Canary-2023-12-13-20-16-49'},
    {'AlarmName': 'TestAlarm-ModelLatency-Final-Deployment-Guardrails-Canary-2023-12-13-20-16-4
9'}]}},
 'ResponseMetadata': {'RequestId': '3d3c6c0c-e16d-4c71-87b2-14460a0fdb8a',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': '3d3c6c0c-e16d-4c71-87b2-14460a0fdb8a',
   'content-type': 'application/x-amz-json-1.1',
```

In [19]:
```
invoke_endpoint(endpoint_name)
```

```
Sending test traffic to the endpoint Final-Deployment-Guardrails-Canary-2023-12-13-20-16-49.
Please wait...
EEEEEEEEE
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[19], line 1
----> 1 invoke_endpoint(endpoint_name)

Cell In[10], line 11, in invoke_endpoint(endpoint_name, max_invocations, wait_interval_sec, should_raise
_exp)
      9 payload = row.rstrip("\n")
     10 try:
---> 11     response = sm_runtime.invoke_endpoint(
     12         EndpointName=endpoint_name, ContentType="text/csv", Body=payload
     13     )
     14     response["Body"].read()
     15     print(".", end="", flush=True)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/client.py:553, in ClientCreator._cre
ate_api_method.<locals>._api_call(self, *args, **kwargs)
    549     raise TypeError(
    550         f"{py_operation_name}() only accepts keyword arguments."
    551     )
    552 # The "self" in this scope is referring to the BaseClient.
--> 553 return self._make_api_call(operation_name, kwargs)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/client.py:989, in BaseClient._make_a
pi_call(self, operation_name, api_params)
    985     maybe_compress_request(
    986         self.meta.config, request_dict, operation_model
    987     )
    988     apply_request_checksum(request_dict)
--> 989     http, parsed_response = self._make_request(
    990         operation_model, request_dict, request_context
    991     )
    993 self.meta.events.emit(
    994     'after-call.{service_id}.{operation_name}'.format(
    995         service_id=service_id, operation_name=operation_name
   (...)
   1000     context=request_context,
   1001 )
   1003 if http.status_code >= 300:

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/client.py:1015, in BaseClient._make_
request(self, operation_model, request_dict, request_context)
   1013 def _make_request(self, operation_model, request_dict, request_context):
   1014     try:
-> 1015         return self._endpoint.make_request(operation_model, request_dict)
   1016     except Exception as e:
   1017         self.meta.events.emit(
   1018             'after-call-error.{service_id}.{operation_name}'.format(
   1019                 service_id=self._service_model.service_id.hyphenize(),
   (...)
   1023             context=request_context,
   1024         )

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/endpoint.py:119, in Endpoint.make_re
quest(self, operation_model, request_dict)
    113 def make_request(self, operation_model, request_dict):
    114     logger.debug(
    115         "Making request for %s with params: %s",
    116         operation_model,
    117         request_dict,
    118     )
--> 119     return self._send_request(request_dict, operation_model)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/endpoint.py:199, in Endpoint._send_r
equest(self, request_dict, operation_model)
    197 self._update_retries_context(context, attempts)
    198 request = self.create_request(request_dict, operation_model)
--> 199 success_response, exception = self._get_response(
    200     request, operation_model, context
    201 )
    202 while self._needs_retry(
    203     attempts,
    204     operation_model,
    (...)
    207     exception,
    208 ):
    209     attempts += 1

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/endpoint.py:241, in Endpoint._get_re
sponse(self, request, operation_model, context)
    235 def _get_response(self, request, operation_model, context):
    236     # This will return a tuple of (success_response, exception)
    237     # and success_response is itself a tuple of
    238     # (http_response, parsed_dict).
    239     # If an exception occurs then the success_response is None.
    240     # If no exception occurs then exception is None.
--> 241     success_response, exception = self._do_get_response(
    242         request, operation_model, context
    243     )
    244     kwargs_to_emit = {
    245         'response_dict': None,
    246         'parsed_response': None,
    247         'context': context,
    248         'exception': exception,
    249     }
    250     if success_response is not None:

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/endpoint.py:281, in Endpoint._do_get
_response(self, request, operation_model, context)
    279     http_response = first_non_none_response(responses)
    280     if http_response is None:
--> 281         http_response = self._send(request)
    282 except HTTPClientError as e:
    283     return (None, e)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/endpoint.py:377, in Endpoint._send(s
elf, request)
    376 def _send(self, request):
--> 377     return self.http_session.send(request)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/botocore/httpsession.py:464, in URLLib3Sessio
n.send(self, request)
    461     conn.proxy_headers['host'] = host
    463 request_target = self._get_request_target(request.url, proxy_url)
--> 464 urllib_response = conn.urlopen(
    465     method=request.method,
    466     url=request_target,
    467     body=request.body,
    468     headers=request.headers,
    469     retries=Retry(False),
    470     assert_same_host=False,
    471     preload_content=False,
    472     decode_content=False,
    473     chunked=self._chunked(request.headers),
    474 )
    476 http_response = botocore.awsrequest.AWSResponse(
    477     request.url,
    478     urllib_response.status,
    479     urllib_response.headers,
    480     urllib_response,
    481 )
    483 if not request.stream_output:
    484     # Cause the raw stream to be exhausted immediately. We do it
    485     # this way instead of using preload_content because
    486     # preload_content will never buffer chunked responses

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/urllib3/connectionpool.py:715, in HTTPConnect
ionPool.urlopen(self, method, url, body, headers, retries, redirect, assert_same_host, timeout, pool_tim
eout, release_conn, chunked, body_pos, **response_kw)
    712     self._prepare_proxy(conn)
    714 # Make the request on the httplib connection object.
--> 715 httplib_response = self._make_request(
    716     conn,
    717     method,
    718     url,
    719     timeout=timeout_obj,
    720     body=body,
    721     headers=headers,
    722     chunked=chunked,
    723 )
    725 # If we're going to release the connection in ``finally:``, then
    726 # the response doesn't need to know about the connection. Otherwise
    727 # it will also try to release it and we'll have a double-release
    728 # mess.
    729 response_conn = conn if not release_conn else None

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/urllib3/connectionpool.py:467, in HTTPConnect
ionPool._make_request(self, conn, method, url, timeout, chunked, **httplib_request_kw)
    462             httplib_response = conn.getresponse()
    463         except BaseException as e:
    464             # Remove the TypeError from the exception chain in
    465             # Python 3 (including for exceptions like SystemExit).
    466             # Otherwise it looks like a bug in the code.
--> 467             six.raise_from(e, None)
    468 except (SocketTimeout, BaseSSLError, SocketError) as e:
    469     self._raise_timeout(err=e, url=url, timeout_value=read_timeout)

File <string>:3, in raise_from(value, from_value)

File ~/anaconda3/envs/python3/lib/python3.10/site-packages/urllib3/connectionpool.py:462, in HTTPConnect
ionPool._make_request(self, conn, method, url, timeout, chunked, **httplib_request_kw)
    459 except TypeError:
    460     # Python 3
    461     try:
--> 462         httplib_response = conn.getresponse()
    463     except BaseException as e:
    464         # Remove the TypeError from the exception chain in
    465         # Python 3 (including for exceptions like SystemExit).
    466         # Otherwise it looks like a bug in the code.
    467         six.raise_from(e, None)

File ~/anaconda3/envs/python3/lib/python3.10/http/client.py:1375, in HTTPConnection.getresponse(self)
   1373 try:
   1374     try:
-> 1375         response.begin()
   1376     except ConnectionError:
   1377         self.close()

File ~/anaconda3/envs/python3/lib/python3.10/http/client.py:318, in HTTPResponse.begin(self)
    316 # read until we get a non-100 response
    317 while True:
--> 318     version, status, reason = self._read_status()
    319     if status != CONTINUE:
    320         break

File ~/anaconda3/envs/python3/lib/python3.10/http/client.py:279, in HTTPResponse._read_status(self)
    278 def _read_status(self):
--> 279     line = str(self.fp.readline(_MAXLINE + 1), "iso-8859-1")
    280     if len(line) > _MAXLINE:
    281         raise LineTooLong("status line")

File ~/anaconda3/envs/python3/lib/python3.10/socket.py:705, in SocketIO.readinto(self, b)
    703 while True:
    704     try:
--> 705         return self._sock.recv_into(b)
    706     except timeout:
    707         self._timeout_occurred = True

File ~/anaconda3/envs/python3/lib/python3.10/ssl.py:1307, in SSLSocket.recv_into(self, buffer, nbytes, f
lags)
   1303     if flags != 0:
   1304         raise ValueError(
   1305           "non-zero flags not allowed in calls to recv_into() on %s" %
   1306           self.__class__)
-> 1307     return self.read(nbytes, buffer)
   1308 else:
   1309     return super().recv_into(buffer, nbytes, flags)

File ~/anaconda3/envs/python3/lib/python3.10/ssl.py:1163, in SSLSocket.read(self, len, buffer)
   1161 try:
   1162     if buffer is not None:
```

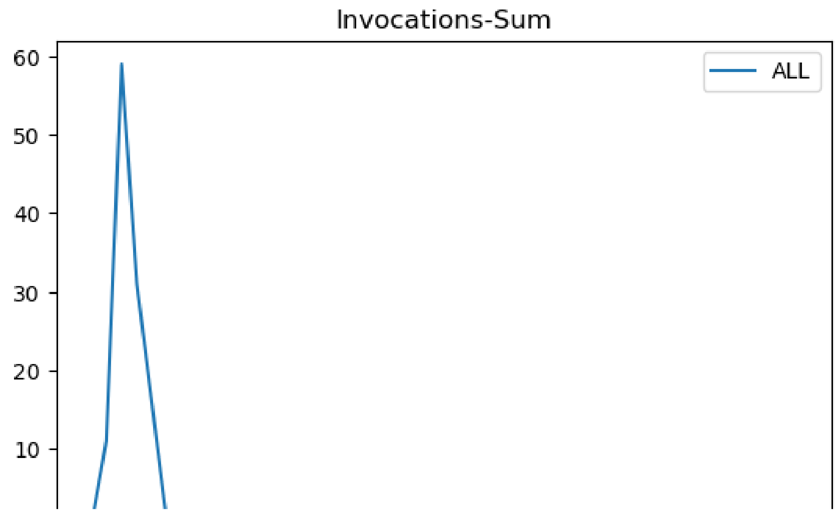
```
-> 1163         return self._sslobj.read(len, buffer)
   1164     else:
   1165         return self._sslobj.read(len)

KeyboardInterrupt:
```

In [20]:
```
wait_for_endpoint_in_service(endpoint_name)

sm.describe_endpoint(EndpointName=endpoint_name)
```

```
Waiting for endpoint in service

Done!
```

Out[20]:
```
{'EndpointName': 'Final-Deployment-Guardrails-Canary-2023-12-13-20-16-49',
 'EndpointArn': 'arn:aws:sagemaker:us-east-1:585522057818:endpoint/final-deployment-guardrails-canary-20
23-12-13-20-16-49',
 'EndpointConfigName': 'Final-EpConfig-2-2023-12-13-20-16-27',
 'ProductionVariants': [{'VariantName': 'AllTraffic',
   'DeployedImages': [{'SpecifiedImage': '785573368785.dkr.ecr.us-east-1.amazonaws.com/image-classificat
ion-neo:latest',
     'ResolvedImage': '785573368785.dkr.ecr.us-east-1.amazonaws.com/image-classification-neo@sha256:85cf
961d93a9ae5583ded12d30612ddc01d1f54d9858ab2857bdb9218de62345',
     'ResolutionTime': datetime.datetime(2023, 12, 13, 20, 26, 59, 301000, tzinfo=tzlocal())}],
   'CurrentWeight': 1.0,
   'DesiredWeight': 1.0,
   'CurrentInstanceCount': 3,
   'DesiredInstanceCount': 3}],
 'EndpointStatus': 'InService',
 'CreationTime': datetime.datetime(2023, 12, 13, 20, 16, 50, 13000, tzinfo=tzlocal()),
 'LastModifiedTime': datetime.datetime(2023, 12, 13, 20, 36, 23, 994000, tzinfo=tzlocal()),
 'LastDeploymentConfig': {'BlueGreenUpdatePolicy': {'TrafficRoutingConfiguration': {'Type': 'CANARY',
    'WaitIntervalInSeconds': 300,
    'CanarySize': {'Type': 'INSTANCE_COUNT', 'Value': 1}},
   'TerminationWaitInSeconds': 120,
   'MaximumExecutionTimeoutInSeconds': 1800},
  'AutoRollbackConfiguration': {'Alarms': [{'AlarmName': 'TestAlarm-5XXErrors-Final-Deployment-Guardrail
s-Canary-2023-12-13-20-16-49'},
    {'AlarmName': 'TestAlarm-ModelLatency-Final-Deployment-Guardrails-Canary-2023-12-13-20-16-49'}]}},
 'ResponseMetadata': {'RequestId': '230d5364-6e16-4e32-8b6a-d35f61a71e24',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': '230d5364-6e16-4e32-8b6a-d35f61a71e24',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '1277',
   'date': 'Wed, 13 Dec 2023 21:06:03 GMT'},
  'RetryAttempts': 0}}
```

In [21]:
```
invocation_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, None, "AllTraffic", "Invocations", "Sum"
)
metrics_epc_1 = plot_endpoint_invocation_metrics(
    endpoint_name, ep_config_name, "AllTraffic", "Invocations", "Sum"
)
metrics_epc_2 = plot_endpoint_invocation_metrics(
    endpoint_name, ep_config_name2, "AllTraffic", "Invocations", "Sum"
)

metrics_all = invocation_metrics.join([metrics_epc_1, metrics_epc_2], how="outer")
metrics_all.plot(title="Invocations-Sum")

invocation_5xx_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, None, "AllTraffic", "Invocation5XXErrors", "Sum"
)
model_latency_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, None, "AllTraffic", "ModelLatency", "Average"
)
```



success case

In [22]:
```
# update endpoint with a valid version of DeploymentConfig

sm.update_endpoint(
    EndpointName=endpoint_name,
    EndpointConfigName=ep_config_name3,
    RetainDeploymentConfig=True,
)
```

Out[22]:
```
{'EndpointArn': 'arn:aws:sagemaker:us-east-1:585522057818:endpoint/final-deployment-guardrails-canary-20
23-12-13-20-16-49',
 'ResponseMetadata': {'RequestId': 'c9e0cf91-b113-4172-89a1-7b7e508552ec',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': 'c9e0cf91-b113-4172-89a1-7b7e508552ec',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '122',
   'date': 'Wed, 13 Dec 2023 21:06:47 GMT'},
  'RetryAttempts': 0}}
```

In [23]:
```
sm.describe_endpoint(EndpointName=endpoint_name)
```

Out[23]:
```
{'EndpointName': 'Final-Deployment-Guardrails-Canary-2023-12-13-20-16-49',
 'EndpointArn': 'arn:aws:sagemaker:us-east-1:585522057818:endpoint/final-deployment-guardrails-canary-20
23-12-13-20-16-49',
 'EndpointConfigName': 'Final-EpConfig-3-2023-12-13-20-16-27',
 'ProductionVariants': [{'VariantName': 'AllTraffic',
   'DeployedImages': [{'SpecifiedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboos
t:1.3-1',
     'ResolvedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboost@sha256:f2e3e321fcf
65be2bb6647c8b0f26439d516447ea0652e8e12fcb767f014c37c',
     'ResolutionTime': datetime.datetime(2023, 12, 13, 21, 12, 19, 17000, tzinfo=tzlocal())}],
   'CurrentWeight': 1.0,
   'DesiredWeight': 1.0,
   'CurrentInstanceCount': 3,
   'DesiredInstanceCount': 3}],
 'EndpointStatus': 'InService',
 'CreationTime': datetime.datetime(2023, 12, 13, 20, 16, 50, 13000, tzinfo=tzlocal()),
 'LastModifiedTime': datetime.datetime(2023, 12, 13, 21, 14, 48, 347000, tzinfo=tzlocal()),
 'ResponseMetadata': {'RequestId': '046b059c-f02d-491d-a9d0-a1ed88657cd5',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': '046b059c-f02d-491d-a9d0-a1ed88657cd5',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '789',
   'date': 'Wed, 13 Dec 2023 21:15:40 GMT'},
  'RetryAttempts': 0}}
```

In [24]:
```
invoke_endpoint(endpoint_name, max_invocations=500)
```

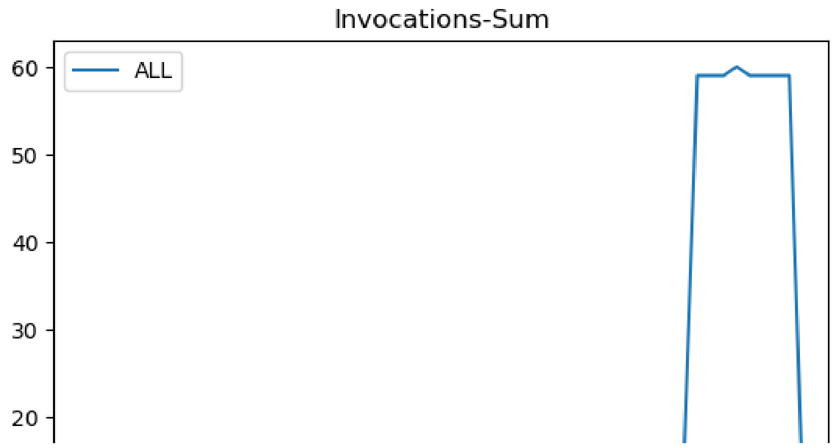
```
Sending test traffic to the endpoint Final-Deployment-Guardrails-Canary-2023-12-13-20-16-49.
Please wait...
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................
Done!
```

In [25]:
```
#wait_for_endpoint_in_service(endpoint_name)

sm.describe_endpoint(EndpointName=endpoint_name)
```

Out[25]:
```
{'EndpointName': 'Final-Deployment-Guardrails-Canary-2023-12-13-20-16-49',
 'EndpointArn': 'arn:aws:sagemaker:us-east-1:585522057818:endpoint/final-deployment-guardrails-canary-20
23-12-13-20-16-49',
 'EndpointConfigName': 'Final-EpConfig-3-2023-12-13-20-16-27',
 'ProductionVariants': [{'VariantName': 'AllTraffic',
   'DeployedImages': [{'SpecifiedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboos
t:1.3-1',
     'ResolvedImage': '683313688378.dkr.ecr.us-east-1.amazonaws.com/sagemaker-xgboost@sha256:f2e3e321fcf
65be2bb6647c8b0f26439d516447ea0652e8e12fcb767f014c37c',
     'ResolutionTime': datetime.datetime(2023, 12, 13, 21, 12, 19, 17000, tzinfo=tzlocal())}],
   'CurrentWeight': 1.0,
   'DesiredWeight': 1.0,
   'CurrentInstanceCount': 3,
   'DesiredInstanceCount': 3}],
 'EndpointStatus': 'InService',
 'CreationTime': datetime.datetime(2023, 12, 13, 20, 16, 50, 13000, tzinfo=tzlocal()),
 'LastModifiedTime': datetime.datetime(2023, 12, 13, 21, 14, 48, 347000, tzinfo=tzlocal()),
 'ResponseMetadata': {'RequestId': 'afa60bb8-f3df-4253-9e21-d71f697d9d74',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': 'afa60bb8-f3df-4253-9e21-d71f697d9d74',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '789',
   'date': 'Wed, 13 Dec 2023 21:27:01 GMT'},
  'RetryAttempts': 0}}
```

In [28]:
```
invocation_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, None, "AllTraffic", "Invocations", "Sum"
)
metrics_epc_1 = plot_endpoint_invocation_metrics(
    endpoint_name, ep_config_name, "AllTraffic", "Invocations", "Sum"
)
metrics_epc_2 = plot_endpoint_invocation_metrics(
    endpoint_name, ep_config_name2, "AllTraffic", "Invocations", "Sum"
)
metrics_epc_3 = plot_endpoint_invocation_metrics(
    endpoint_name, ep_config_name3, "AllTraffic", "Invocations", "Sum"
)

metrics_all = invocation_metrics.join([metrics_epc_1, metrics_epc_2, metrics_epc_3], how="outer")
metrics_all.plot(title="Invocations-Sum")

invocation_5xx_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, None, "AllTraffic", "Invocation5XXErrors", "Sum"
)
model_latency_metrics = plot_endpoint_invocation_metrics(
    endpoint_name, None, "AllTraffic", "ModelLatency", "Average"
)
```

```
AttributeError: 'NoneType' object has no attribute 'index'
```



Cleanup

In [58]:
```
sm.delete_endpoint(EndpointName=endpoint_name)
```

Out[58]:
```
{'ResponseMetadata': {'RequestId': 'b9a3f959-1443-4b89-afe3-3023158b57ae',
  'HTTPStatusCode': 200,
  'HTTPHeaders': {'x-amzn-requestid': 'b9a3f959-1443-4b89-afe3-3023158b57ae',
   'content-type': 'application/x-amz-json-1.1',
   'content-length': '0',
   'date': 'Wed, 13 Dec 2023 01:12:07 GMT'},
  'RetryAttempts': 0}}
```

In [2]:
```
sm.delete_endpoint_config(EndpointConfigName=ep_config_name)
sm.delete_endpoint_config(EndpointConfigName=ep_config_name2)
sm.delete_endpoint_config(EndpointConfigName=ep_config_name3)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[2], line 1
----> 1 sm.delete_endpoint_config(EndpointConfigName=ep_config_name)
      2 sm.delete_endpoint_config(EndpointConfigName=ep_config_name2)
      3 sm.delete_endpoint_config(EndpointConfigName=ep_config_name3)

NameError: name 'sm' is not defined
```

In [41]:
```
# sm.delete_model(ModelName=model_name)
# sm.delete_model(ModelName=model_name2)
# sm.delete_model(ModelName=model_name3)
```

In [42]:
```
# cw.delete_alarms(AlarmNames=[error_alarm, latency_alarm])
```

Shadow Testing

In [34]:
```
response = sm.create_inference_experiment(
    Name='Final-project-employees-shadow',
    Type='ShadowMode',
    Description='string',
    RoleArn=f'{role}',
    EndpointName='endpoint-shadow-config',
    ModelVariants=[
        {
            'ModelName': model_name3,
            'VariantName': 'production-variant',
            'InfrastructureConfig': {
                'InfrastructureType': 'RealTimeInference',
                'RealTimeInferenceConfig': {
                    'InstanceType': 'ml.t2.xlarge',
                    'InstanceCount': 1
                }
            }
        },
        {
            'ModelName': model_name,
            'VariantName': 'shadow-variant',
            'InfrastructureConfig': {
                'InfrastructureType': 'RealTimeInference',
                'RealTimeInferenceConfig': {
                    'InstanceType': 'ml.t2.xlarge',
                    'InstanceCount': 1
                }
            }
        },
    ],
    ShadowModeConfig={
        'SourceModelVariantName': 'production-variant',
        'ShadowModelVariants': [
            {
                'ShadowModelVariantName': 'shadow-variant',
                'SamplingPercentage': 100
            },
        ]
    },
)
```

In [ ]:
```
def invoke_endpoint(endpoint_name, should_raise_exp=False):
    with open("test_data/test.csv", "r") as f:
        for row in f:
            payload = row.rstrip("\n")
            try:
                for i in range(10):  # send the same payload 10 times for testing purpose
                    response = sm_runtime.invoke_endpoint(
                        EndpointName=endpoint_name, ContentType="text/csv", Body=payload
                    )
                print(".", end="", flush=True)
            except Exception as e:
                print(e)
                print("E", end="", flush=True)
                if should_raise_exp:
                    raise e

invoke_endpoint("endpoint-shadow-config")
```

```
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
```

In [44]:
```
invocation_metrics = plot_endpoint_invocation_metrics(
 endpoint_name, None, "production", "Invocations", "Sum"
)
invocation_4xx_metrics = plot_endpoint_invocation_metrics(
 endpoint_name, None, "AllTraffic", "Invocation4XXErrors", "Sum"
)
invocation_5xx_metrics = plot_endpoint_invocation_metrics(
 endpoint_name, None, "AllTraffic", "Invocation5XXErrors", "Sum"
)
model_latency_metrics = plot_endpoint_invocation_metrics(
 endpoint_name, None, "production", "ModelLatency", "Average"
)
model_latency_metrics = plot_endpoint_invocation_metrics(
 endpoint_name, None, "shadow", "ModelLatency", "Average"
)
```

In [51]:
```python
# delete shadow test and endpoint configuration
sm_client = boto3.client('sagemaker')

# Specify the names of your shadow endpoint and endpoint configuration
shadow_endpoint_name = 'your-shadow-endpoint-name'
shadow_endpoint_config_name = 'your-shadow-endpoint-config-name'

# Delete the shadow endpoint
sm_client.delete_endpoint(EndpointName=shadow_endpoint_name)

# Delete the shadow endpoint configuration
sm_client.delete_endpoint_config(EndpointConfigName=shadow_endpoint_config_name)

# Optionally, delete the model associated with the shadow endpoint
model_name = 'your-model-name'
sm_client.delete_model(ModelName=model_name)
```

In [ ]: