SageMaker Batch Transform

Generate large offline predictions with an Sklearn example

Ram Vegiraju
Published in

Towards Data Science

.
5 min read
.
Apr 25, 2022
76
1

Image from Unsplash by Jonathon Farber

In my last article I talked about the latest SageMaker Inference option in Serverless Inference. An older, yet equally important option is SageMaker Batch Transform. Sometimes for our Machine Learning models we don't necessarily need a persistent endpoint. We just have a **large set of data** and we want **inference** returned for that data. This is a great option for workloads that **don't have** any **latency requirements** and are purely focused with returning inference on a dataset

Using SageMaker Batch Transform we'll explore how you can take a **Sklearn regression model** and get **inference** on a **sample dataset**. The dataset in this example is not necessarily large, but in

reality you can use Batch Transform for thousands of data points. Some very popular use cases include preprocessing of a dataset before you need real-time inference or working with image data for Computer Vision workloads.

NOTE: For those of you new to AWS, make sure you make an account at the following **link** if you want to follow along. There will be costs incurred through the deployment process for the Batch Transform Job. This article will also assume intermediate knowledge of SageMaker and AWS.

Table of Contents

Dataset

For our example we'll be working with the Petrol Consumption regression dataset from Kaggle. The original data source is licensed here.

Setup

For our example we'll be working with the Petrol Consumption regression dataset from Kaggle. Before we can get to inference, we'll be training a Sklearn Random Forest Model using SageMaker. In the script we'll also have custom inference handlers to be able to work with the csv dataset that we feed in for inference. We won't be walking through the entire training setup in depth in this example, but you can read an end to end guide right here.

For this example, we'll be working in SageMaker Studio with a Data Science Kernel and ml.c5.large instance. You can also use Classic Notebook Instances or your local environment if you've setup your AWS credentials. First we'll read the dataset and take a look at what we're working with.

Read dataset

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	Petrol_Consumption
0	9.0	3571	1976	0.525	541
1	9.0	4092	1250	0.572	524
2	9.0	3865	1586	0.580	561
3	7.5	4870	2351	0.529	414
4	8.0	4399	431	0.544	410

Dataset head (Screenshot by Author)

We'll split this dataset into two portions, one for training and a test set for Batch Inference.

Split dataset

We can then push this data to S3 as usual where SageMaker will grab the training data and dump the model artifacts along with the inference output later.

Upload data to S3

Another set of code you'll find in the notebook is that you can **locally test Batch Inference** without SageMaker first. Take the model that you want to work with and train it locally on the train.csv file. After that take the model artifact (joblib file for Sklearn) and perform inference with the test.csv file. If you can perform Batch Inference locally then you understand how to adjust your inference handler functions in your train script. This is a **great way to save time** and **debug before** getting to **SageMaker** Training or Inference.

Now we can create our Sklearn estimator that automatically pulls the Amazon supported image for Sklearn. Here we can feed in our training script with our model and inference handlers.

Sklearn estimator

Training Script

```
2022-04-25 02:45:20,324 sagemaker-containers INFO     Reporting training SUCCESS

2022-04-25 02:45:39 Uploading - Uploading generated training model
2022-04-25 02:45:39 Completed - Training job completed
Training seconds: 72
Billable seconds: 72
```

Successful training (Screenshot by Author)

In the next section we'll take a look at the inference function for handling input.

Batch Transform Example

In the training script you'll notice the **input_fn** has been configured for handling a **CSV input**. For examples you may create on your own, this is the function that needs to be adjusted for what input your **model** is **expecting**. In this case we'll feed in a CSV so we configure the input handler for that.

Input function

After our training has completed we can now get to the Batch Inference portion. With **Batch Inference** we do not work with endpoints as the other three SageMaker Inference options do. Here we instantiate a **Transformer** object that will start a Batch Transform job with the parameters you provide. Similar to Real-Time Inference we can grab the trained estimator and create a transformer off of it.

Transformer

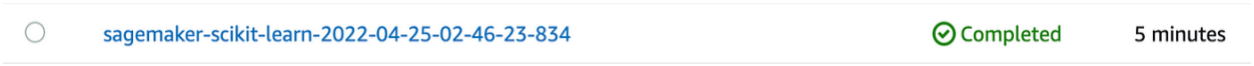
Even though we don't cover it in this article, the two knobs you can adjust to optimize Batch Inference are: **max_concurrent_transforms** and **max_payload**. With max payload you can control the **input payload size** and with concurrent transforms you can control the number **parallel requests** that can be sent to each instance in a transform job. By default they are set to the values shown in the screenshot below.

```
169.254.255.130 - - [25/Apr/2022:02:50:54 +0000] "POST /invocations HTTP/1.1" 200 76 "-" "Go-http-cl
ient/1.1"
2022-04-25T02:50:54.774:[sagemaker logs]: MaxConcurrentTransforms=1, MaxPayloadInMB=6, BatchStrategy
=MULTI_RECORD
```

Batch Job Succeeds (Screenshot by Author)

We can now execute our Transform job and you can also monitor this through the SageMaker Console.

Transform Job



Console (Screenshot by Author)

After the job has completed, the **results** will be dumped to an **S3 location**. We can grab that S3 URI using the [Boto3 client] for SageMaker and parse the results file to display our outputs.

Parse Transform Results (Screenshot by Author)

	599.4
0	593.25
1	560.70
2	619.40
3	580.75
4	636.30
5	567.45
6	561.20
7	531.60
8	494.65
9	609.90
10	538.00

Transform Results (Screenshot by Author)

Additional Resources & Conclusion

SageMaker-Deployment/BatchTransform/BYOM-Sklearn at master · RamVegiraju/SageMaker-Deployment
Compilation of examples of SageMaker inference options and other features. …
github.com

For the entire **code** for the example access the link **above**. Batch Inference has a great number of use-cases and I hope this article serves as a good primer and reference for your to try out this Inference option with your own models and workloads. Check out another cool Batch Inference example with HuggingFace at the following link. If you're more into video tutorials here's a great Batch Transform Section in the following course.

As always I hope this was a good article for you with SageMaker Inference, feel free to leave any feedback or questions in the comments. If you're interested in more AWS/SageMaker related content check out this list that I have compiled for you.

If you enjoyed this article feel free to connect with me on LinkedIn and subscribe to my Medium Newsletter. If you're new to Medium, sign up using my Membership Referral.