Solution: Triggering a Lambda Function

The code solution is located in `Lesson 3, Exercise 2 - Invoking Lambda Functions Solution.ipynb` in the course Github repo.

A small error in the demo: at 1:17, the instructor said "how to invoke an s3 function asynchronously". It should be a Lambda function.

Triggering from CLI

Below is a valid CLI function that will invoke a lambda function.

```
aws lambda invoke --function-name preprocess-helloblze --payload '{"s3-dataset-uri": "udacity-sagemaker-solutiondata2021/l3e1/reviews_Musical_Instruments_5.json.zip"}' response.json
```

Let's break this command down:

- `aws lambda` is the service we're using.
- `--function-name` is the name of the function we're invoking.
- `--payload` is the payload we want to send to the function.
- `response.json` is where we want the output of this function to be written to.

Triggering from an S3 Upload.

We will need to perform the following steps before modifying our lambda code:

1. Create a new s3 folder within an existing bucket.
2. Create a new lambda trigger for S3, specifying the bucket, specify the folder using the prefix, and specify a suffix of ".zip" to ensure that recursive calls don't occur.

We then need to modify the lambda handler starter code so that it properly parses the event that's sent to it.

To test, we'll upload `reviews_Patio_Lawn_and_Garden_5.json.zip` in this directory to your S3 bucket. To see if the lambda function is triggered, you can go to the Monitor tab.

Code

```
import json
import urllib
from HelloBlazePreprocessLambda import preprocess

def lambda_handler(event, context):
    for r in event['Records']:
        bucket = r['s3']['bucket']['name']
        key = urllib.parse.unquote_plus(r['s3']['object']['key'], encoding='ut
        uri = "/".join([bucket, key])
        preprocess(uri)
    return {
        'statusCode': 200,
        'body': "Good to go!"
    }
```

Note: make sure you stop or delete your notebook instances and other resources to avoid cost!