**Notebook: Time-Series Forecasting, Exercise**

Next, you'll approach the task of time-series forecasting. You'll be taking a look at household energy consumption data, originally taken from [Kaggle](https://www.kaggle.com/uciml/electric-power-consumption-data-set). As you follow along with this lesson, you should work in the referenced SageMaker notebooks. We will present a solution to you, but please try to work on a solution of your own, when prompted. Much of the value in this experience will come from experimenting with the code, **in your own way**.

To open this notebook:

* *Navigate to your SageMaker notebook instance, in the*[*SageMaker console*](https://console.aws.amazon.com/sagemaker/)*, which has been linked to the main*[*Github exercise repository*](https://github.com/udacity/ML_SageMaker_Studies)
* *Activate the notebook instance (if it is in a "Stopped" state), and open it via Jupyter*
* *Click on the exercise notebook in the Time\_Series\_Forecasting directory.*

You may also directly view the exercise and solution notebooks via the repository at the following links:

* [Exercise notebook](https://github.com/udacity/ML_SageMaker_Studies/blob/master/Time_Series_Forecasting/Energy_Consumption_Exercise.ipynb)
* [Solution notebook](https://github.com/udacity/ML_SageMaker_Studies/blob/master/Time_Series_Forecasting/Energy_Consumption_Solution.ipynb)

**The solution notebook is meant to be consulted if you are stuck or want to check your work.**

**Notebook Outline**

We'll go over the following steps to complete the notebook.

* Load in and explore household energy consumption data
* Clean the data and transform it to prepare for training a model
* Format the data into JSON Lines
* Train a DeepAR model on defined context and prediction data points
* Evaluate the model by comparing known and predicted consumption values

**Later: Delete Resources**

At the end of this exercise, and intermittently, you will be reminded to delete your endpoints and resources so that you do not incur any extra processing or storage fees!

# Splitting in Time

We'll evaluate our model on a test set of data. For machine learning tasks like classification, we typically create train/test data by randomly splitting examples into different sets. For forecasting it's important to do this train/test split in **time** rather than a random split of all data points.

## Training Time Series

In general, we can create training data by taking each of our complete time series and leaving off the last prediction\_length data points to create corresponding, training time series.

In code this looks like this:

**def** **create\_training\_series**(complete\_time\_series, prediction\_length):

'''Given a complete list of time series data, create training time series.

:param complete\_time\_series: A list of all complete time series.

:param prediction\_length: The number of points we want to predict.

:return: A list of training time series.

'''

*# get training series*

time\_series\_training = []

**for** ts **in** complete\_time\_series:

*# truncate trailing `prediction\_length` pts*

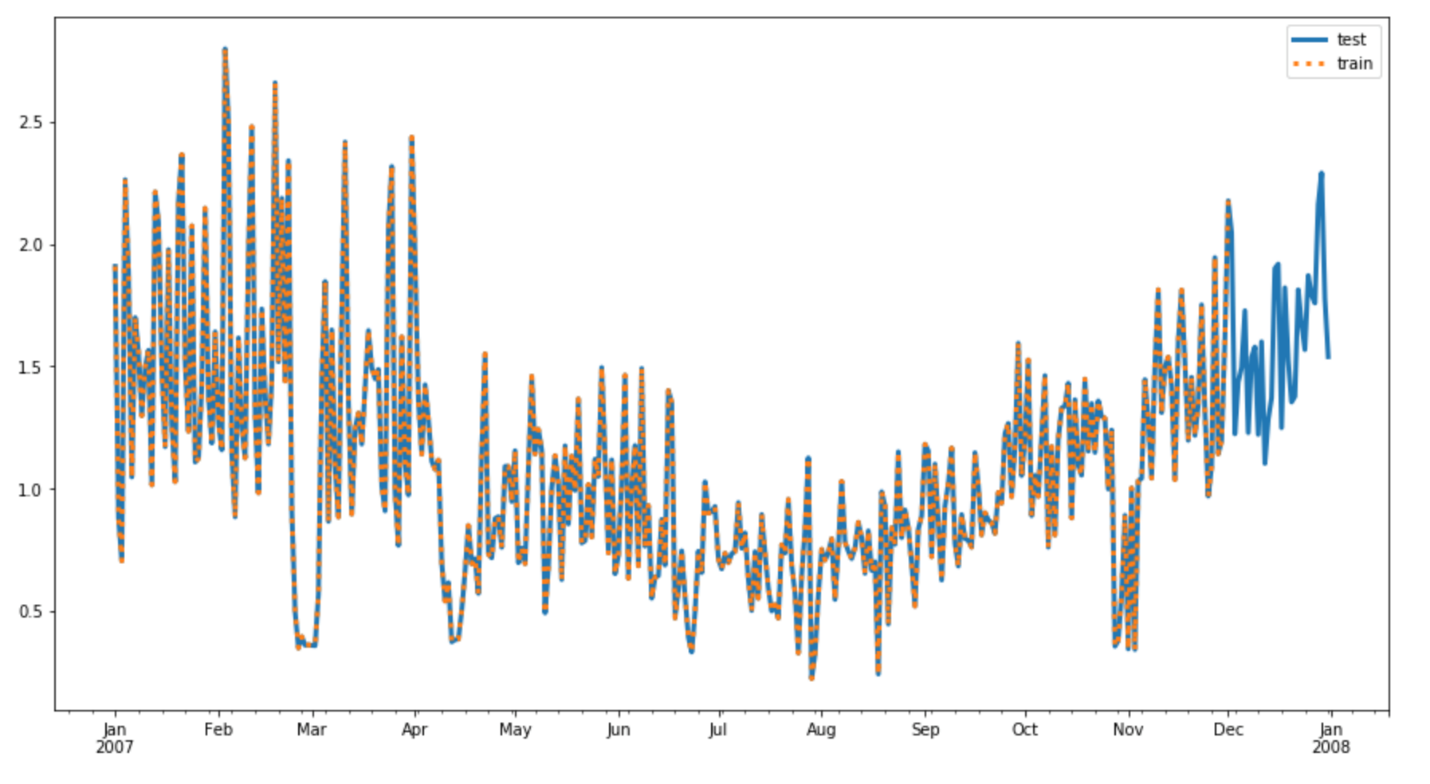
time\_series\_training.append(ts[:-prediction\_length])

**return** time\_series\_training

DeepAR will train on the provided data looking at different intervals that are context\_length number of points as input and the next prediction\_length number of points as output. It selects the context from the given, truncated training data, which is why it is important to leave off the last prediction\_length points.

## Training and Test Series

We can visualize what these series look like, by plotting the train/test series on the same axis. We should see that the test series contains all of our data in a year, and a training series contains all but the last prediction\_length points. Below are train/test series for 2007.



Test series and train series (truncated, in orange).

NEXT

## Instantiating a DeepAR Estimator

Some estimators have specific, SageMaker constructors, but not all. Instead, you can create a base Estimator and pass in the specific **image** (or container) that holds a specific model. The container for the DeepAR model can be gotten as follows:

**from** sagemaker.amazon.amazon\_estimator **import** get\_image\_uri

image\_name = get\_image\_uri(boto3.Session().region\_name, *# get the region*

'forecasting-deepar') *# specify image*

Now that you have the correct image, you can instantiate an Estimator. You're given the following, in-notebook exercise.

### EXERCISE: Instantiate an Estimator

A generic Estimator will be defined by the usual constructor arguments and an image\_name.

You can take a look at the [estimator source code](https://github.com/aws/sagemaker-python-sdk/blob/master/src/sagemaker/estimator.py#L601) to view specifics.

If you complete this task, you can move on to setting DeepAR's model and training hyperparameters and call .fit() to start a training job! You're encouraged to keep going, deploying and evaluating the model on your owm; you are welcome to consult the solution videos to see if your answer matches mine.

NEXT

# Predicting the Future

Recall that we did not give our model any data about 2010, but let's see if it can predict the energy consumption given **no target**, only a known start date!

### EXERCISE: Format a request for a "future" prediction

Your task is to create a formatted input to send to the deployed predictor passing in my usual parameters for "configuration". The "instances" will, in this case, just be one instance, defined by the following:

* **start**: The start time will be time stamp that you specify. To predict the first 30 days of 2010, start on Jan. 1st, '2010-01-01'.
* **target**: The target will be an empty list because this year has no, complete associated time series; we specifically withheld that information from our model, for testing purposes. For example:

{"start": start\_time, "target": []} *# empty target*

You'll see the following code to complete in the main exercise notebook. Complete the instances and see if you can generate some future predictions. **Also, remember to delete your model endpoint when you are done making predictions and evaluating your model.**

*# Starting my prediction at the beginning of 2010*

start\_date = '2010-01-01'

timestamp = '00:00:00'

*# formatting start\_date*

start\_time = start\_date +' '+ timestamp

*# formatting request\_data*

*## TODO: fill in instances information*

request\_data = {"instances": [{"start": **None**, "target": **None**}],

"configuration": {"num\_samples": 50,

"output\_types": ["quantiles"],

"quantiles": ['0.1', '0.5', '0.9']}

}

json\_input = json.dumps(request\_data).encode('utf-8')

print('Requesting prediction for '+start\_time)

**Clean up Resources**

It is good practice to always clean up and delete any resources that you are no longer using. That is, after you complete an exercise, and you are done with predictions and data analysis, you should get rid of any:

* Data source in S3 that you are no longer using
* Endpoint configuration files that you no longer need
* Endpoints that you will no longer use
* CloudWatch logs that are no longer useful

**Deleting Endpoints**

In the notebook, we have usually included code to delete your endpoints after creating some predictions, for example:

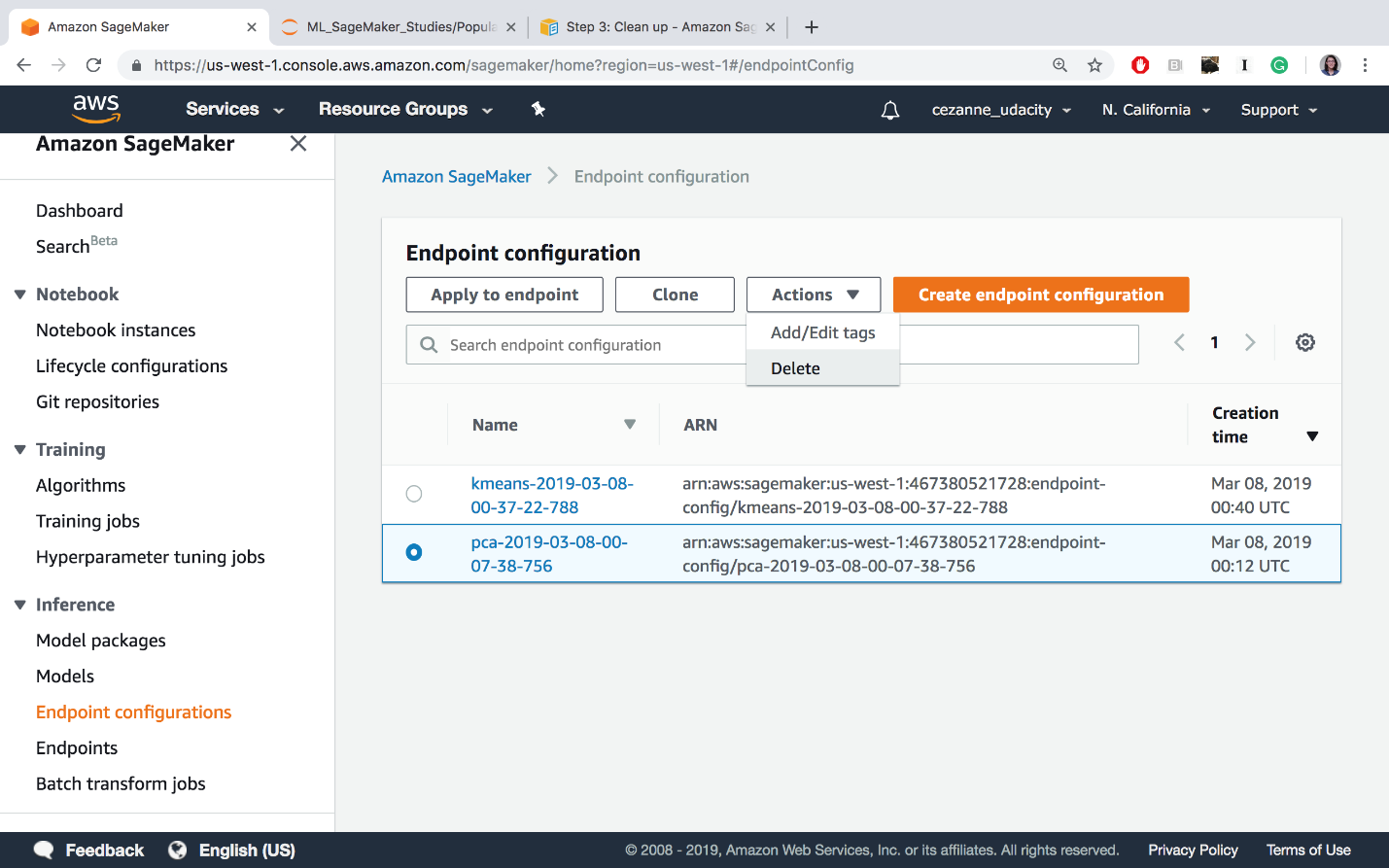
*# delete predictor endpoint*

session.delete\_endpoint(predictor.endpoint)

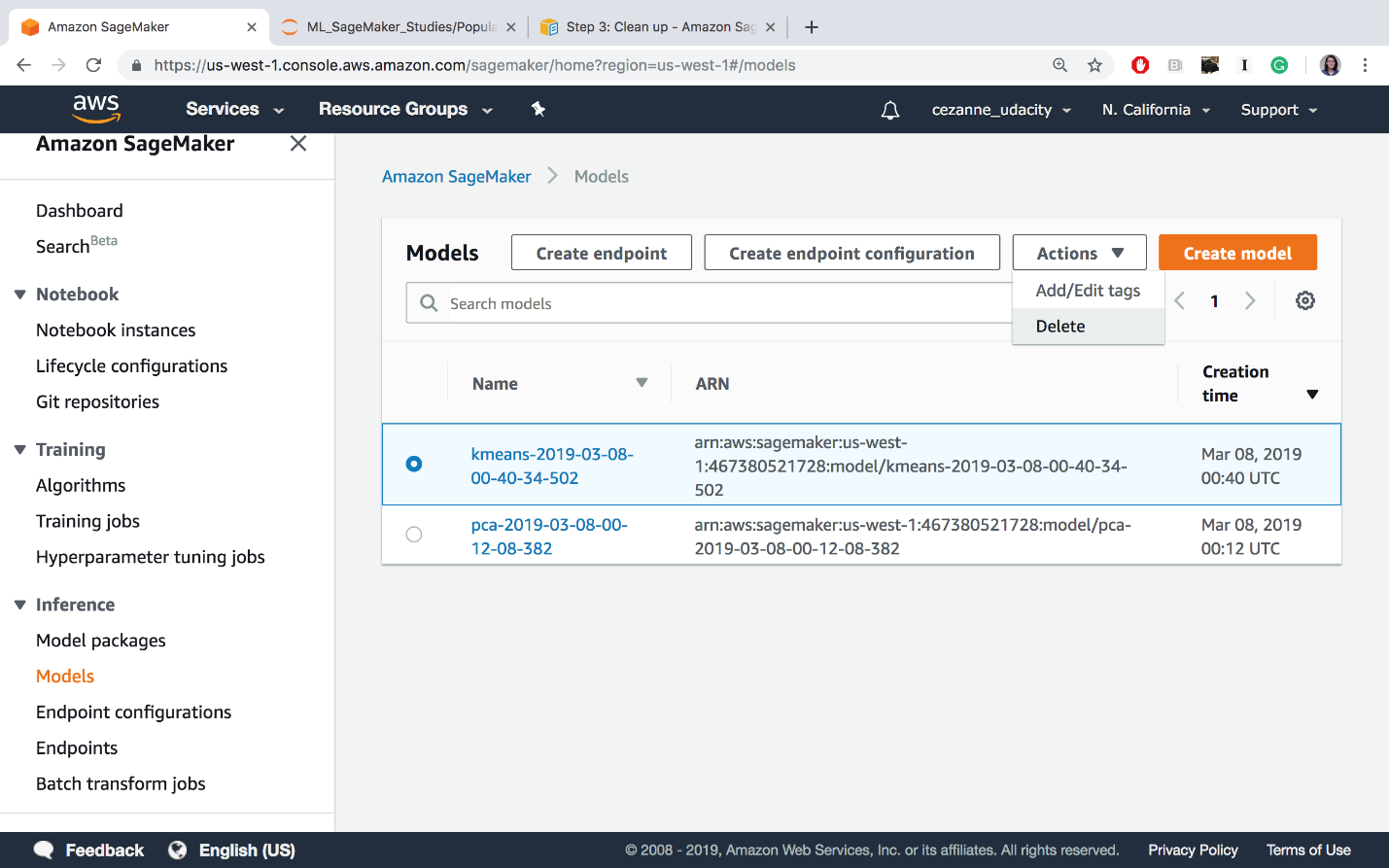
**Thorough Clean up**

You can find a link for instructions on cleaning up all your resources, [in this documentation](https://docs.aws.amazon.com/sagemaker/latest/dg/ex1-cleanup.html) and I will go over some of these details, next.

* Open the Amazon SageMaker console at <https://console.aws.amazon.com/sagemaker/> and delete the following resources:
* The endpoint configuration.
* The model.

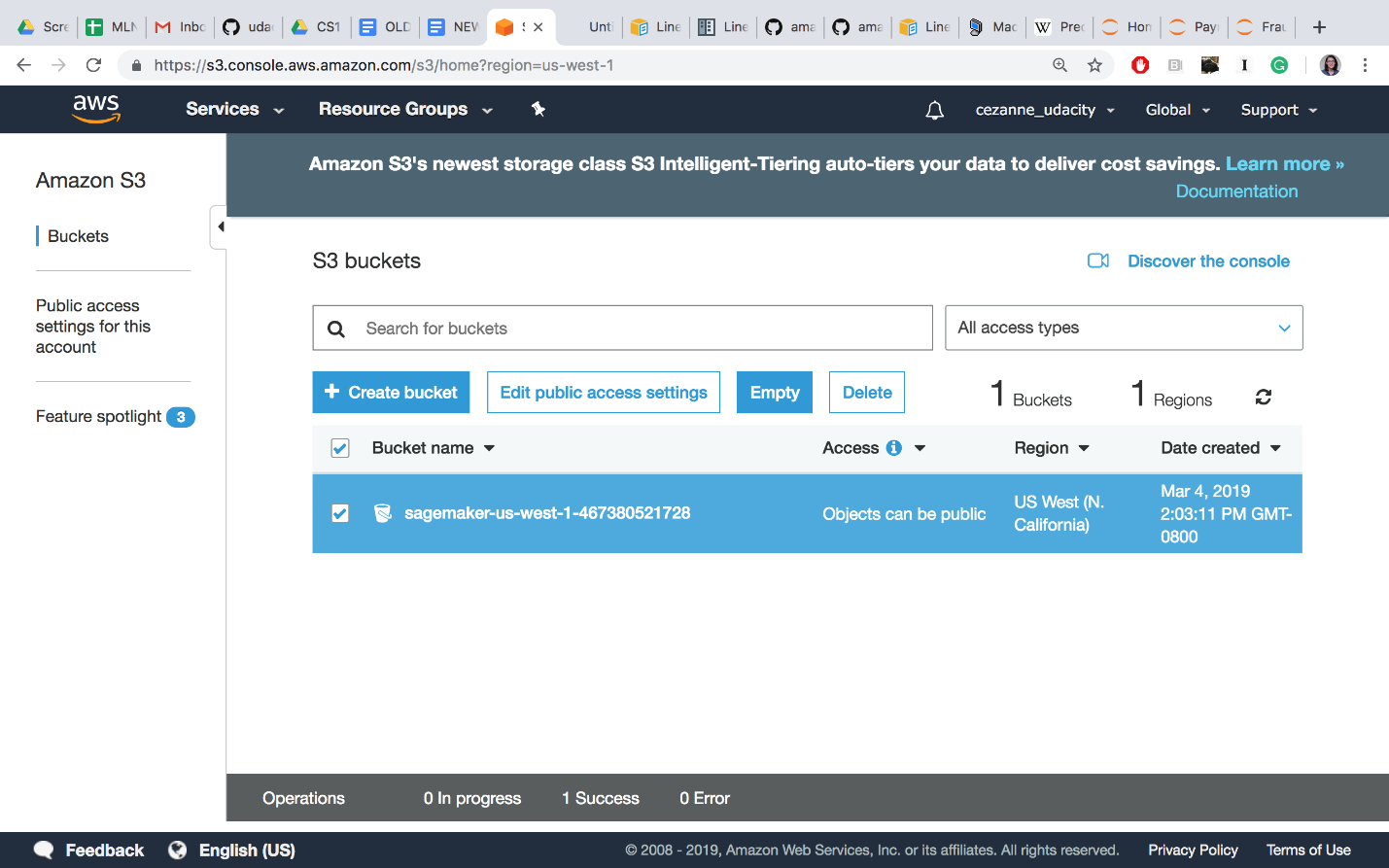


Delete endpoint config files.



Deleting models

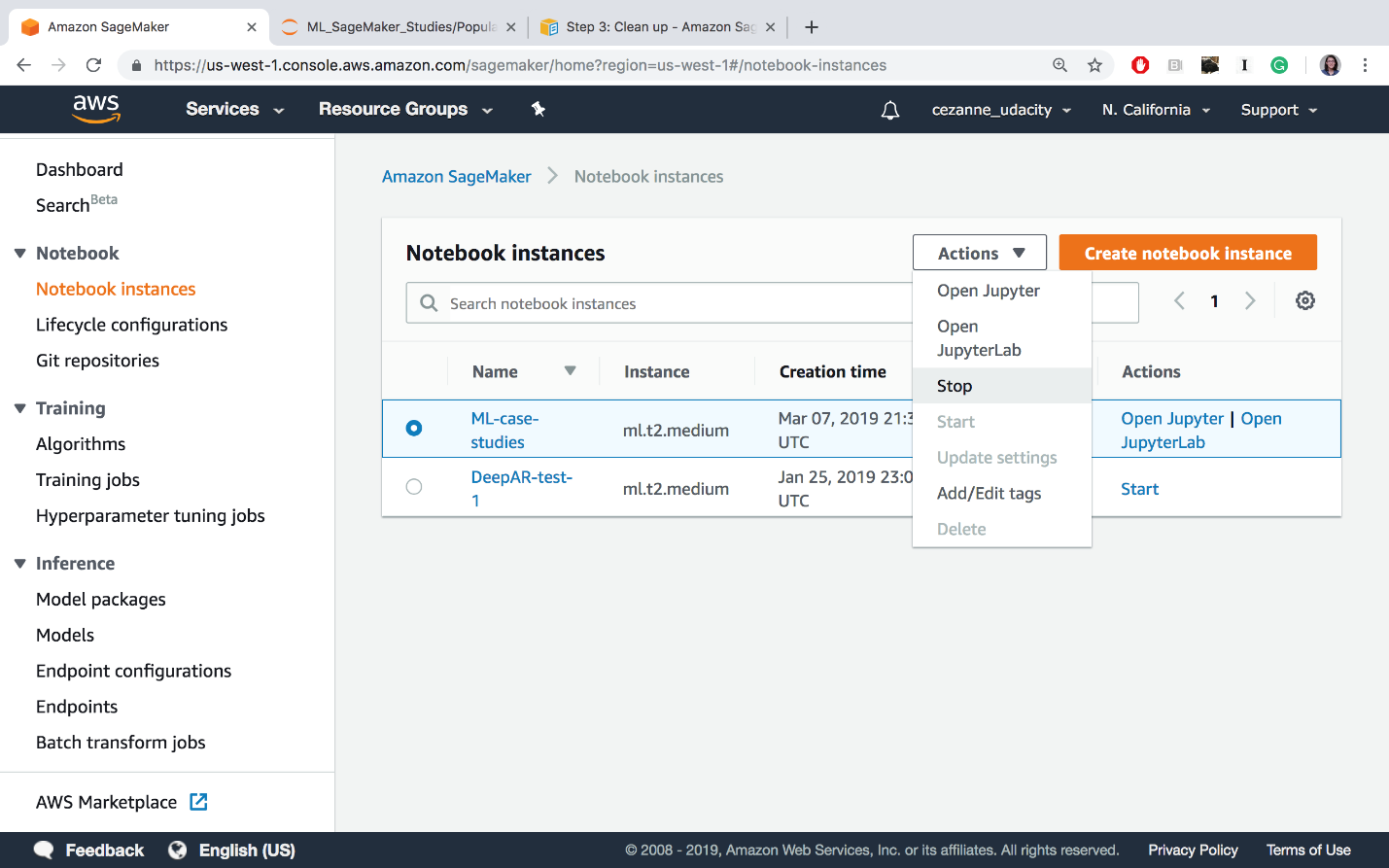
* Open the Amazon S3 console at <https://console.aws.amazon.com/s3/> and delete or empty the bucket that you created for storing model artifacts and the training dataset.



Delete or empty your S3 bucket (empty is recommended until the end of the course, when you should delete this bucket entirely)

* Open the Amazon CloudWatch console at <https://console.aws.amazon.com/cloudwatch/> and delete all of the log groups that have names starting with /aws/sagemaker/.

At the end of this course, you may also choose to delete the entire notebook instance and IAM Role, but you may keep these as is, for now. In between lessons, if you are taking a break, you may want to **Stop** your notebook and pause it from continuously running.



Stopping the ML-case-studies notebook

Cleaning up resources at the end of an exercise or lesson is a great practice to get into!

**IMPORTANT**

**To avoid incurring additional charges, it is suggested that you *DELETE* any unused notebooks and data resources on S3 and CloudWatch.**

NEXT