SageMaker Jumpstart

Use MyApps

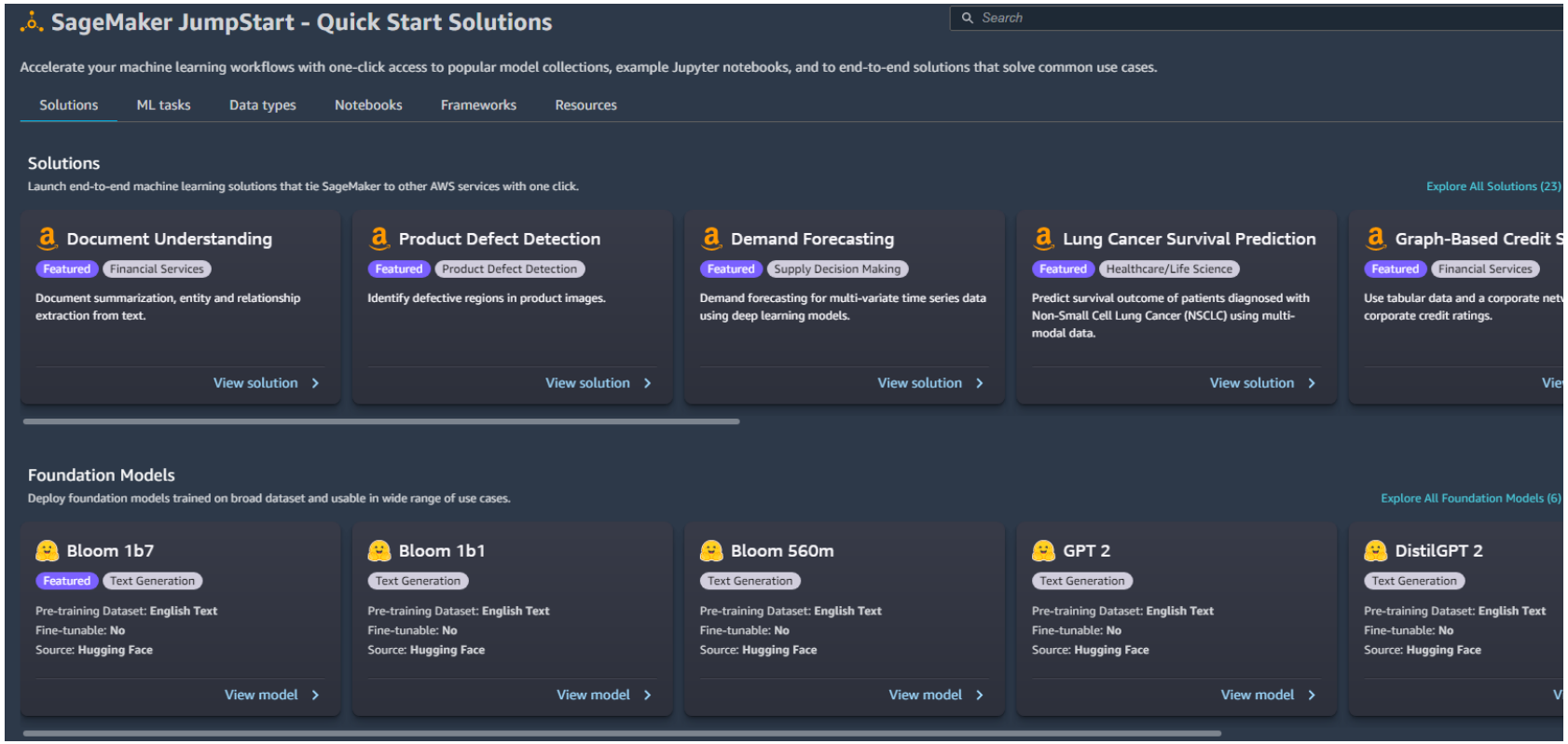
In ca-central

What is SageMaker Jumpstart

- It is a service that has a set of best practices to create an ML solution
- There are more than a dozen of solutions available in Jumpstart
- The use cases are based on multiple industries like manufacturing, retail and finance

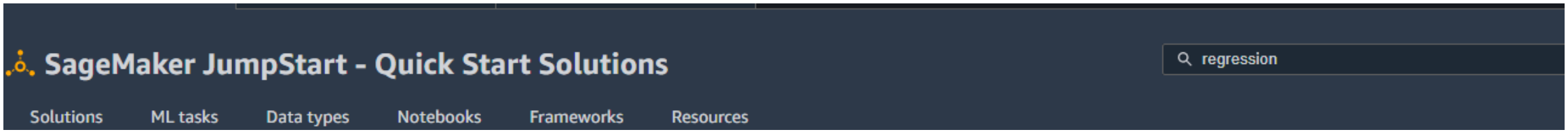
Interface to work with JumpStart

- Solutions based on ML tasks, Data types, notebooks, frameworks



We want to try a regression model

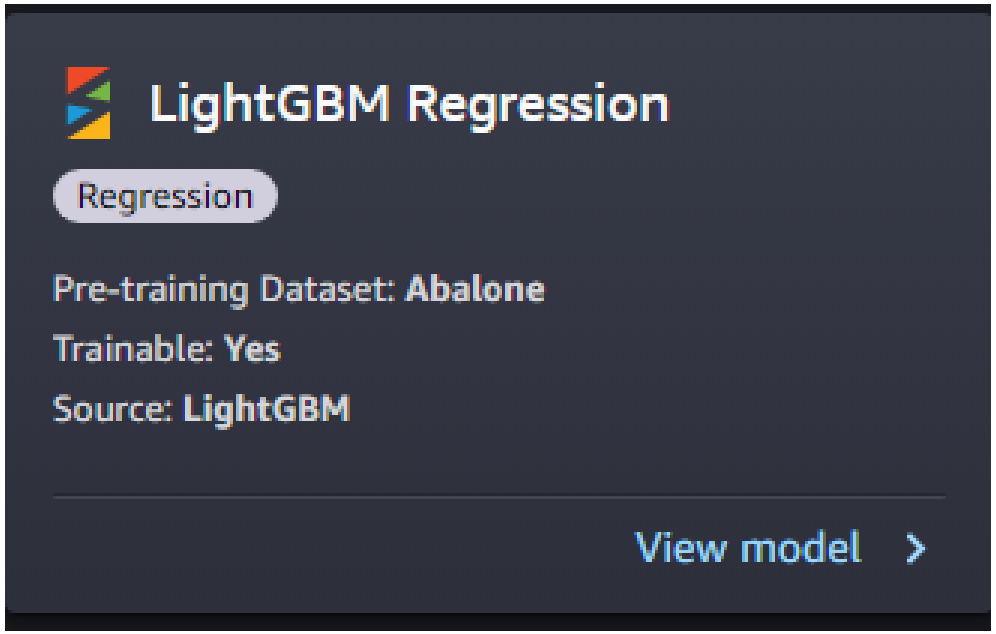
- Search regression in the SageMaker Jumpstart



- Select **Regression with XGBoost** → That is the just to see what's inside that notebook. The notebook is about estimating Abalone age by the data from UCI data repository
- But we do not use this data and instead we use a data set to predict the **likelihood of acceptance for university**

LightGBM Regression

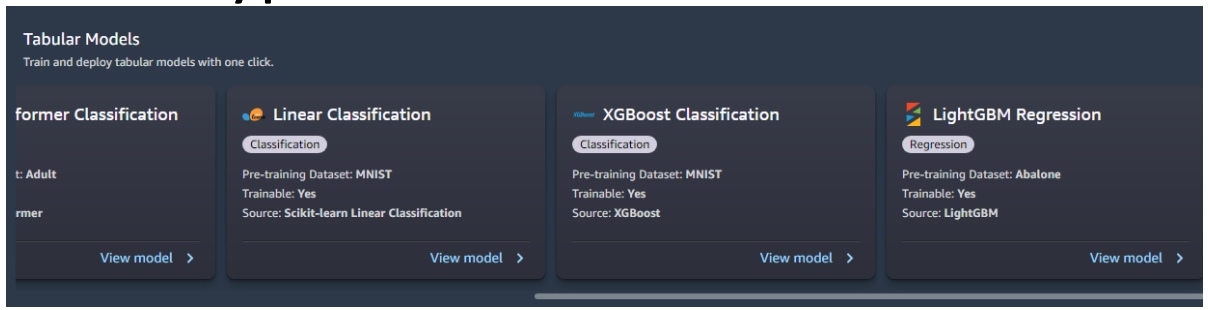
- On the same page you also see a model that is pre-trained on the Abalone data set



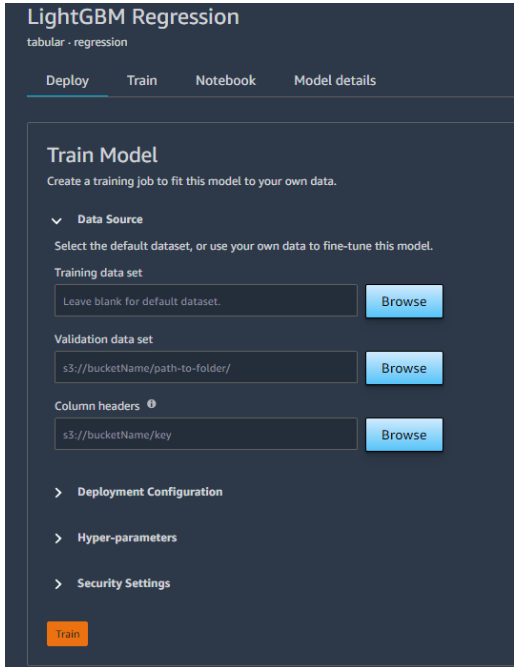
- Click on the LightGBM regression tile

LightGBM Regression

- Alternatively: In StudioJS → data types → Tabular and select **LightGBM Regression**:



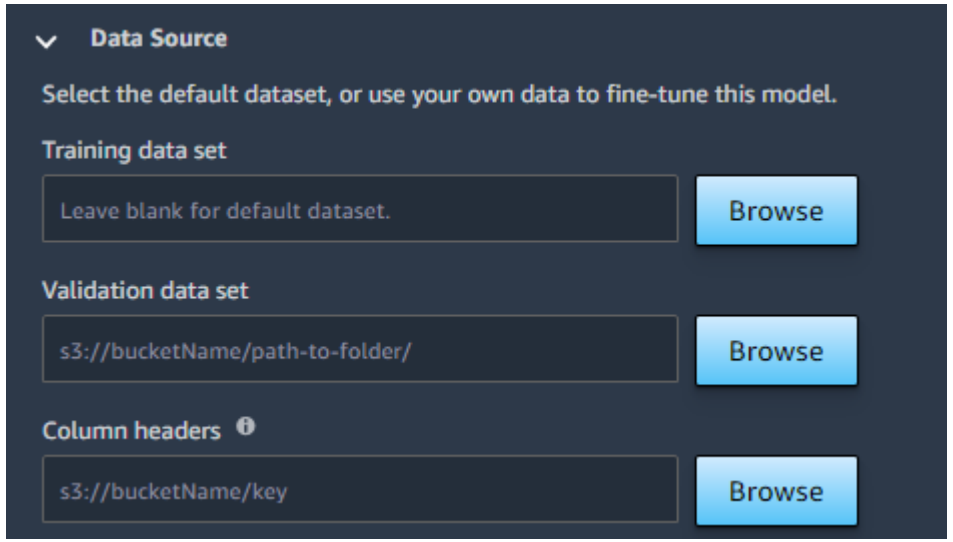
- We can set the parameters in the JS and just click on Train



Introduction to the data set

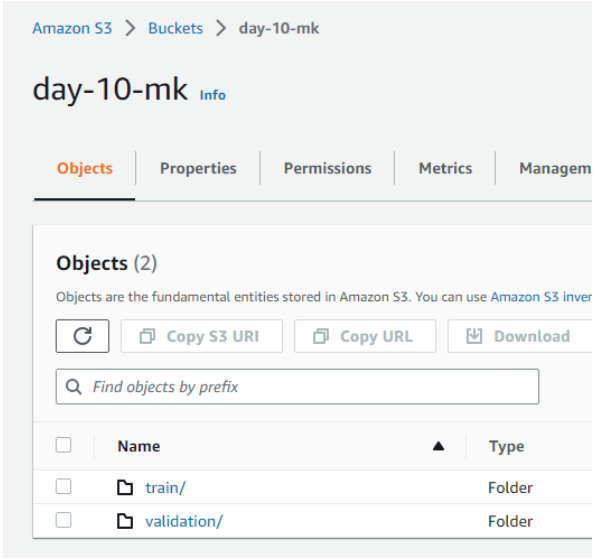
- We want to use a data set to create a model to estimate the chance of admission in university based on GRE, TOEFL and other criterial

	A	B	C	D	E	F	G	H	I
1	GRE_Score	TOEFL_Sc	University_	SOP	LOR	CGPA	Research	Chance_of_Admission	
2	337	118	4	4.5	4.5	9.65	1	0.92	
3	324	107	4	4	4.5	8.87	1	0.76	
4	316	104	3	3	3.5	8	1	0.72	



Upload Dataset

- I have uploaded 3 data files for university admission (training, validation and test data sets)
- Upload the training and validation files in **two folders** folders in a bucket (you need to create a bucket). The name of folders must be **train** and **validation**.



Rename the training dataset

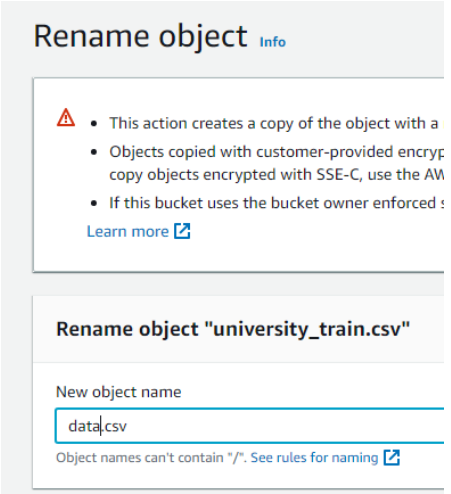
- In the notebook it is assumed that the training and validation dataset names are **data.csv**

Train the Model on a New Dataset

Below are the instructions for how the training data should be formatted for input to the model.

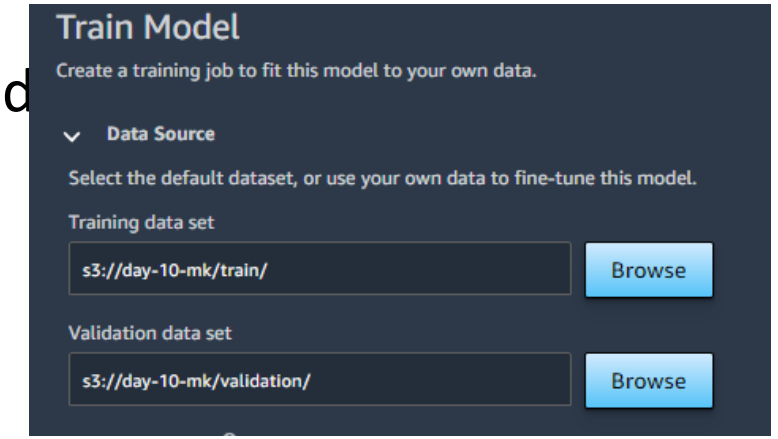
- **Input:** A directory containing two sub-directories 'train/', 'validation/' (optional), and a json-format file named 'categorical_index.json' (optional). Each sub-directory contains a 'data.csv' file.
  - The 'data.csv' files under sub-directory 'train/' and 'validation/' are for training and validation, respectively. The validation data is used to compute a validation score at the end of each boosting iteration. An early stopping is applied when the validation score stops improving. If the validation data is not provided, a 20% of training data is randomly sampled to serve as the validation data.
  - The first column of the 'data.csv' should have the corresponding target variable. The rest of other columns should have the corresponding predictor variables (features).

- Rename the datasets to data.csv in both folders



Configure the jumpstart

- Copy the train and validation folder in the respective field
- NOTE: You need to put **bucket name/foldername**.



- In **Deployment Configuration:** Change the instance type to smaller instance like m5.xlarge
- Review hyperparameters
- In Security Settings:
  - Change the IAM role by selecting **Input IAM Role**
  - Change it to something like**:**

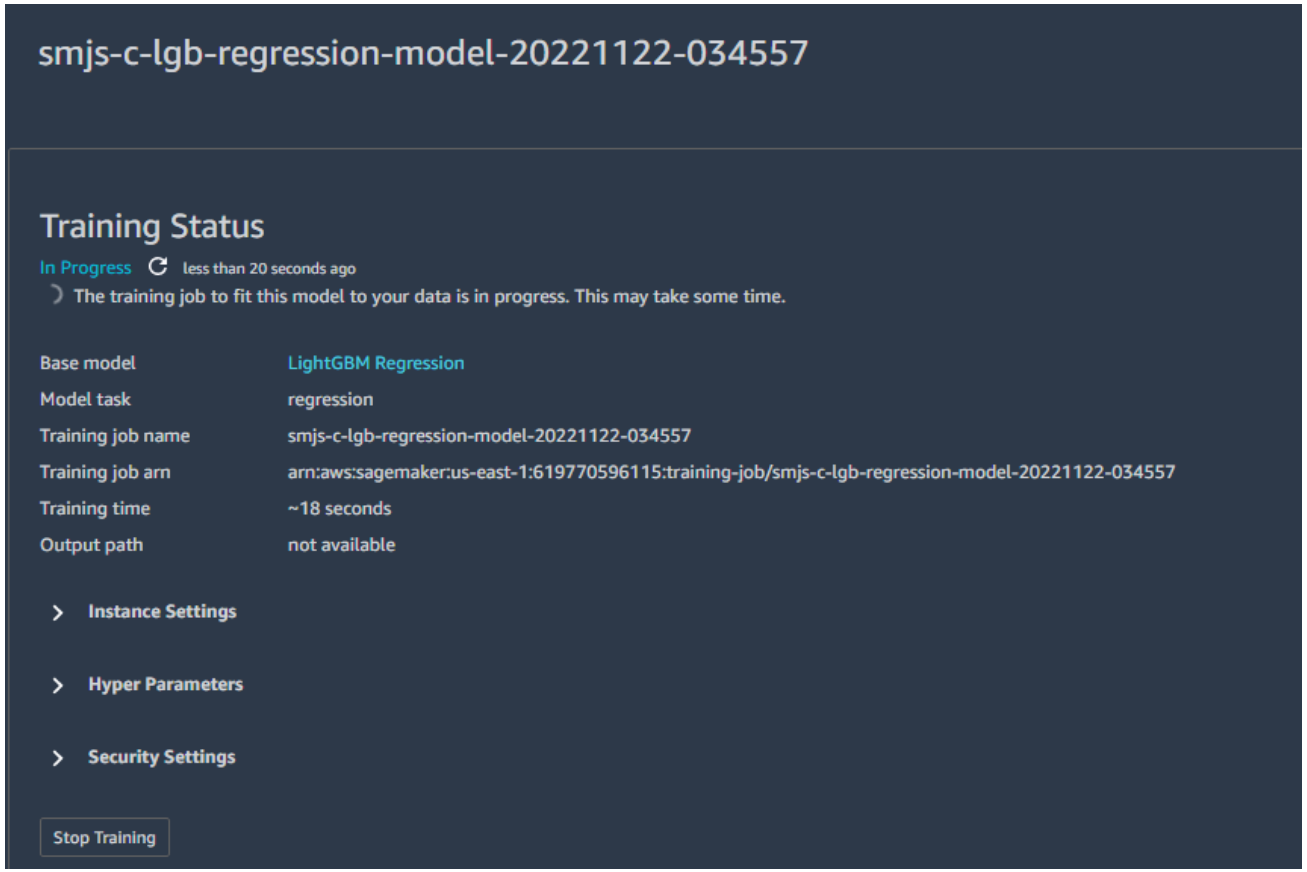
arn:aws:iam::239630988601:role/fast-ai-academic-1-Student-Azure

  - **Change the red items according to your account info**

Start training

- Click on Train
- After a few minutes the

Model is ready



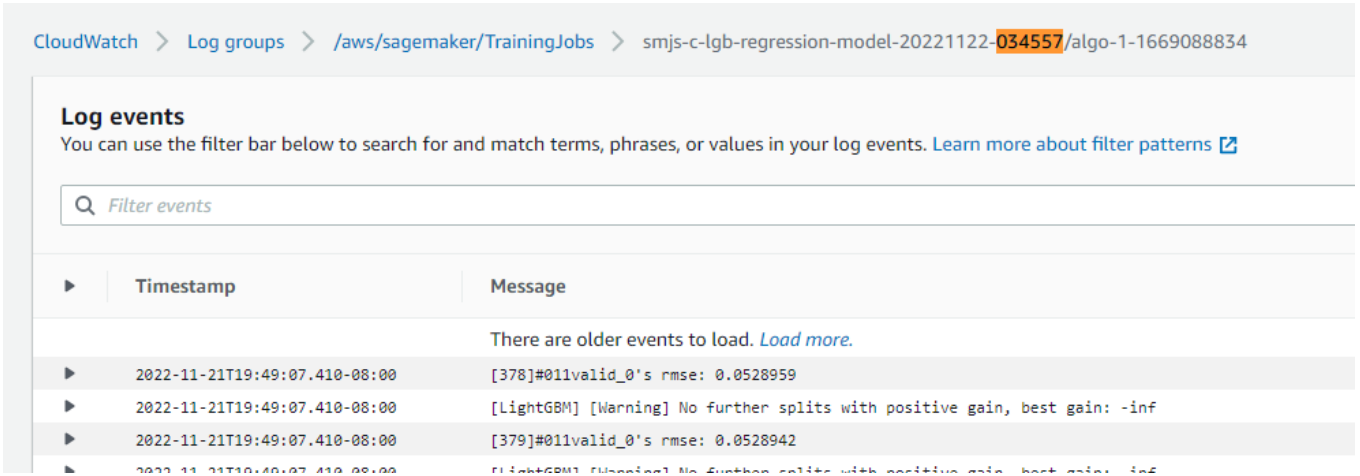
You can also go to SM console and see that the training has started:

Evaluate the metrics of the model

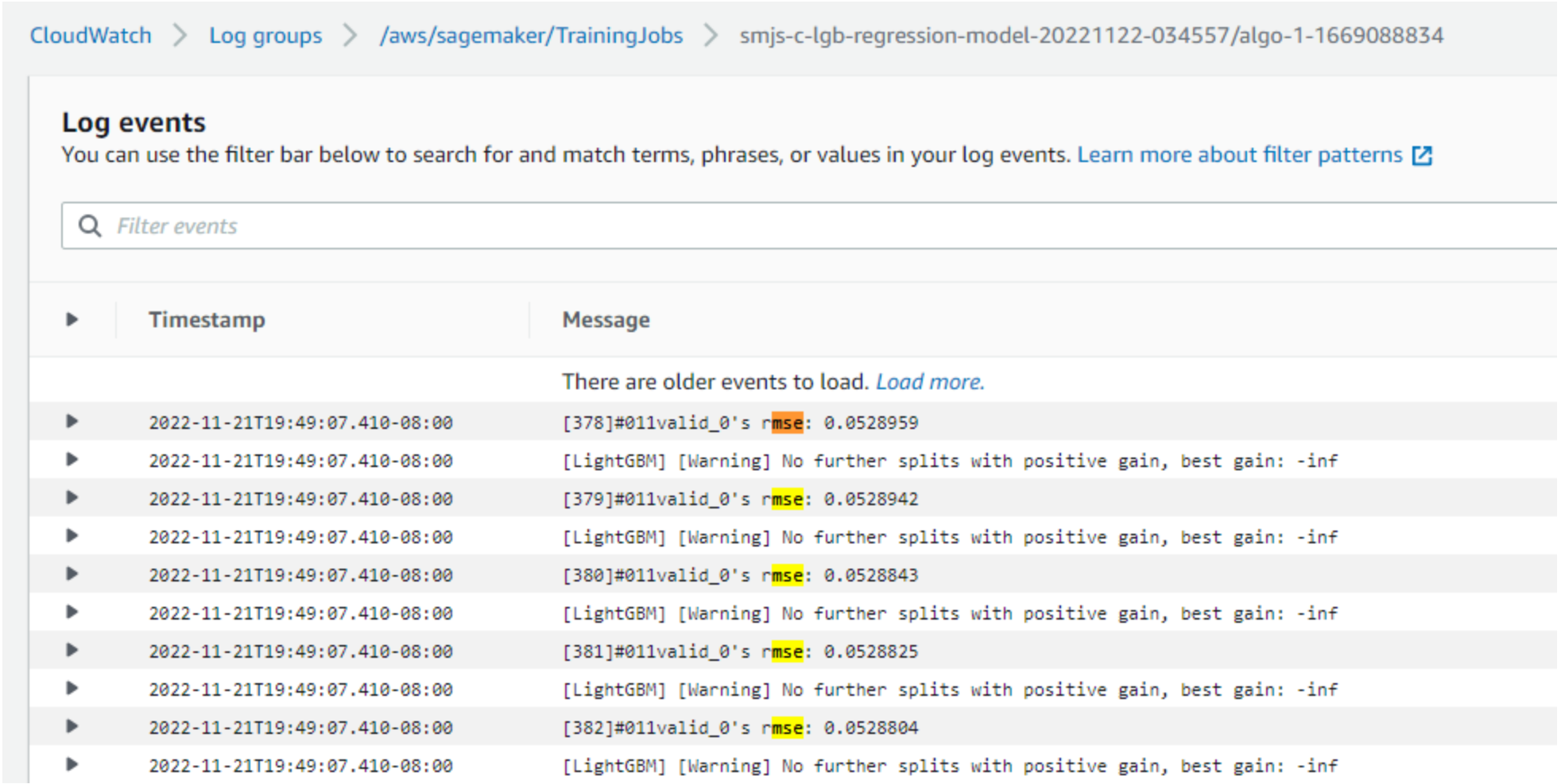
- Grab the training job name:

```
smjs-c-lgb-regression-model-20221122-034557
```

- Search in CloudWatch, the log group and in the **TrainingJob** folder search for the training job that ends with the number of your job name:

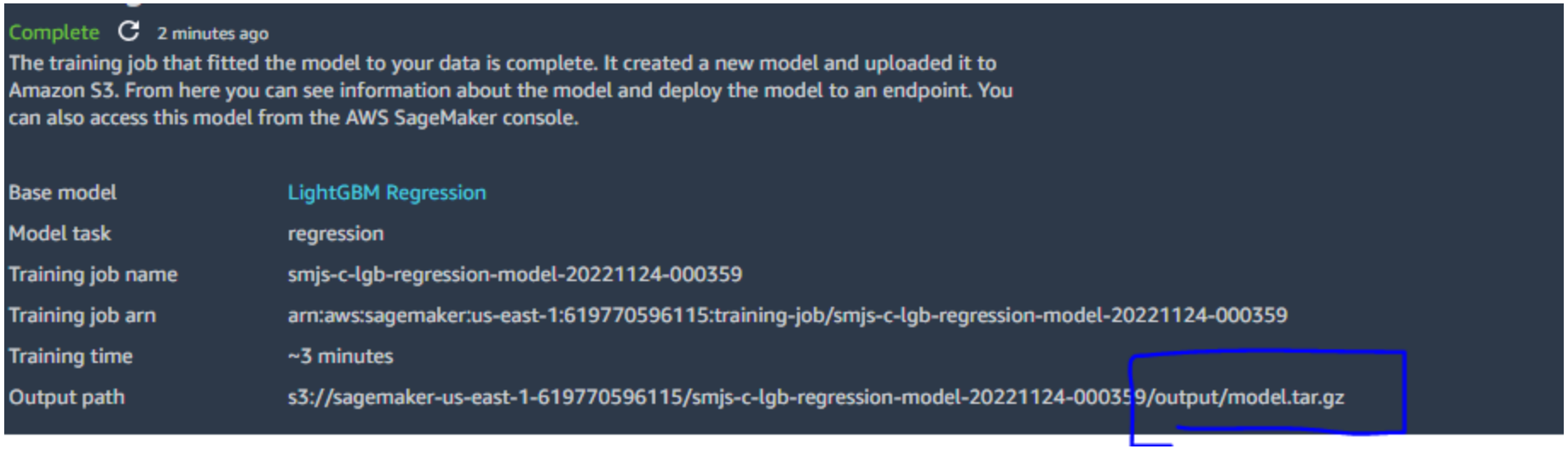


Compare training and validation RMSEs



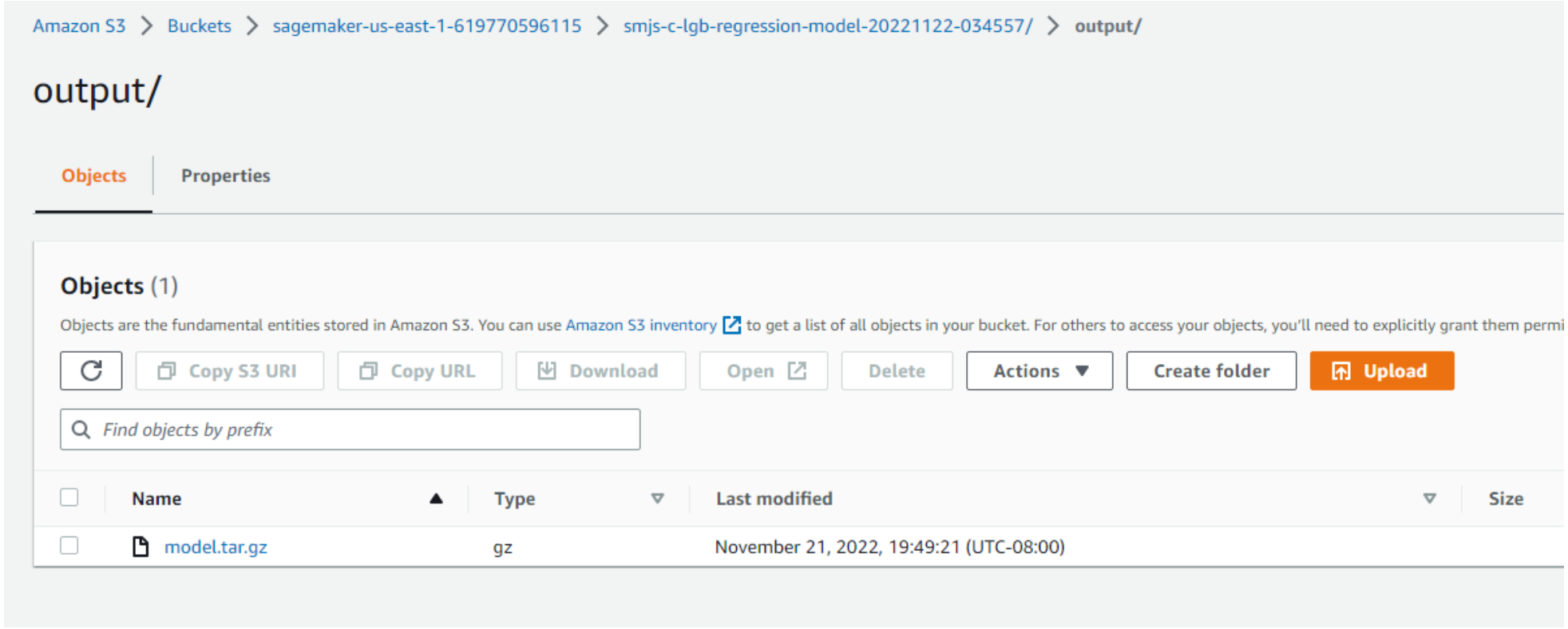
After completion…

- You also should know where the model is: in the path in front of output path



Where is the model

- The model itself is in the S3 bucket:



Deploy the model

- Now that we have the model, we use JS to deploy the model
- Set the instance size and S3 bucket in

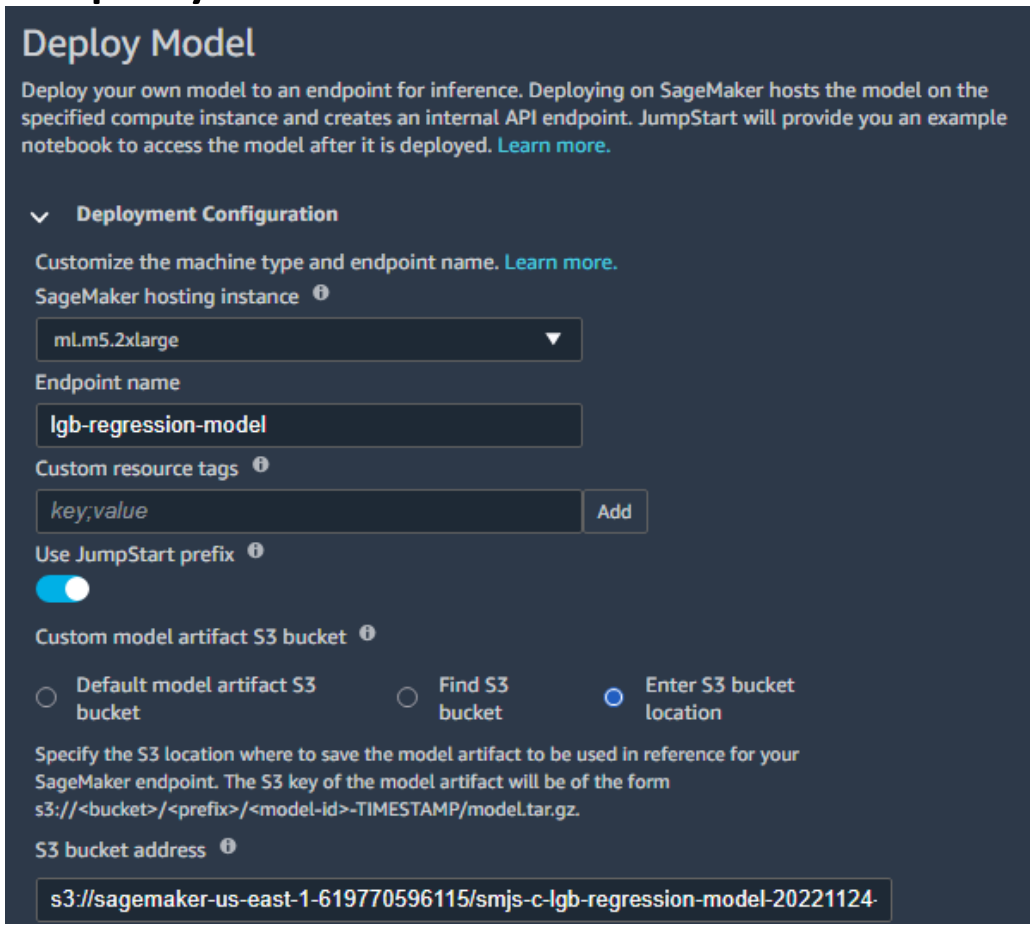
Deployment Configuration

- Set the S3 path by selecting

“Enter S3 Bucket location”

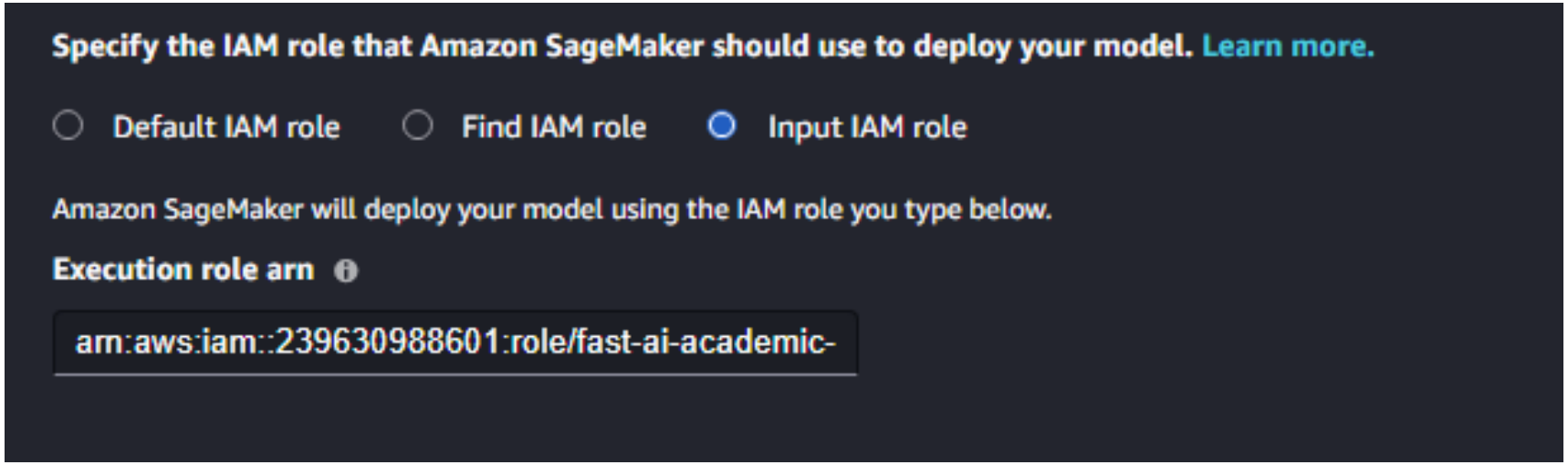
and paste the model.tar.gz name

(object **S3 URI**)



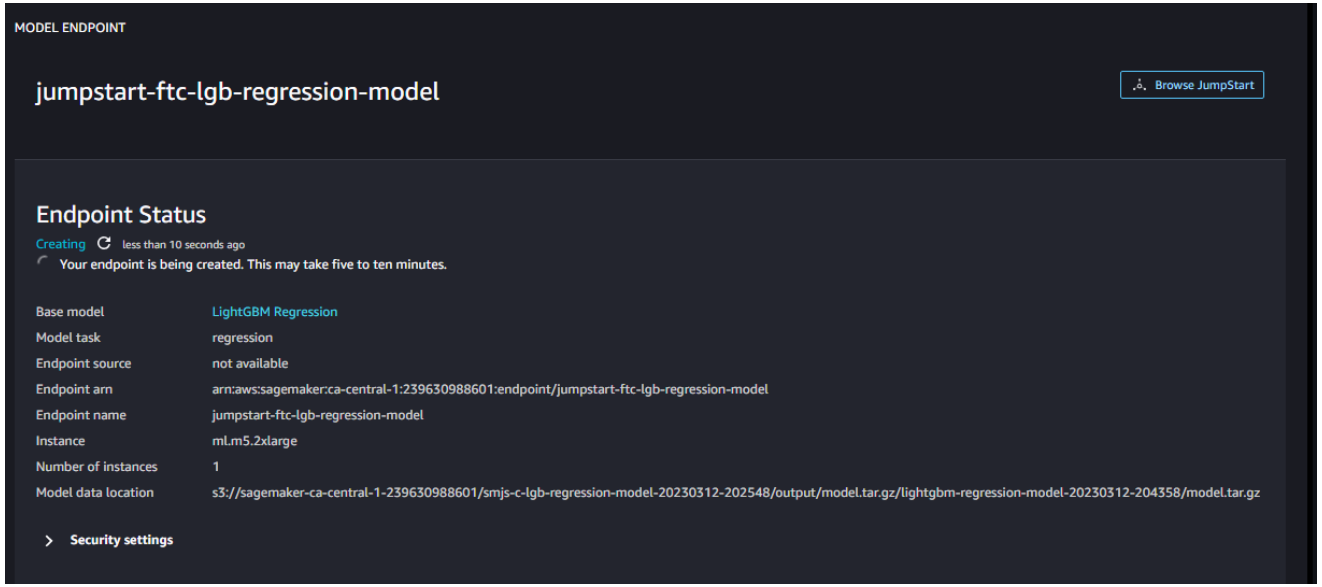
Security settings

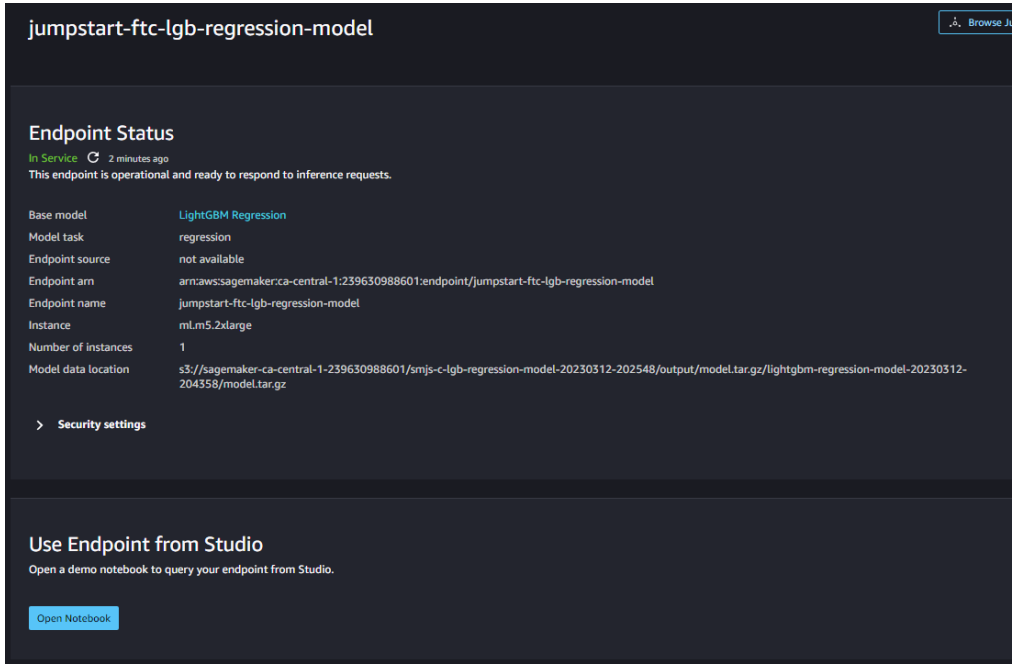
- Set the security setting as shown below:



Deploying the model

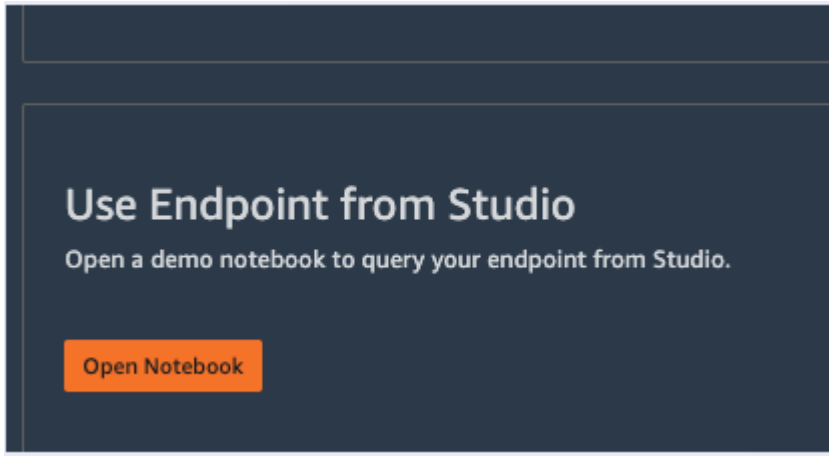
- Click on Deploy and wait until it deploys the endpoint





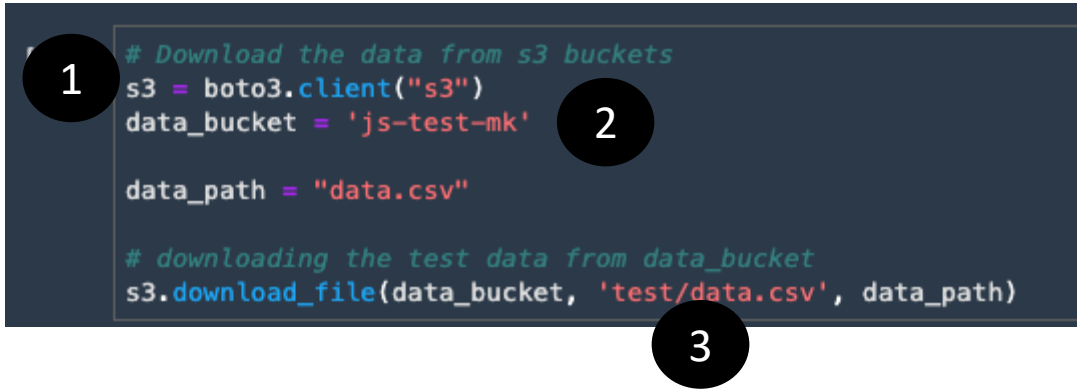
Prepare for inferencing

- Upload the test data to the same bucket but a new folder called **test**
- Make sure the object name is **data.csv**
- Click on **Open Notebook**, in the Use Endpoint from Studio



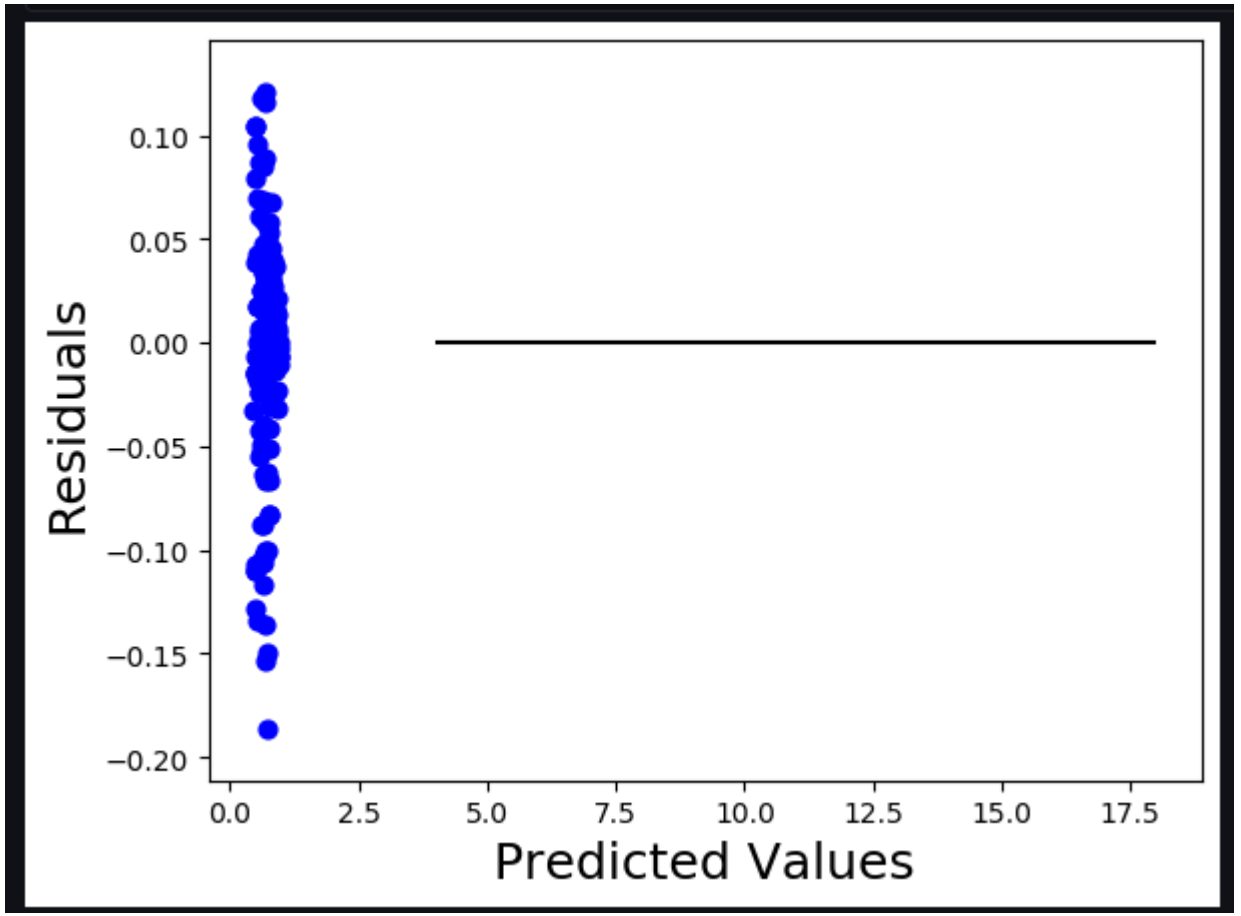
After deploying, test the endpoint

- In the notebook, change the bucket to where the test data is located on
- Make sure the object name is **data.csv** in the data bucket, and apply the changes as shown in the picture
- You have to change 3 spots in the cell
- Make sure you do not put extra / or anything else that is not needed



You see now the metric against the test data

- As you see the residual of the resulting model was pretty good



Delete the endpoint

- Before deleting the endpoint, read the assignment page in the next slide, capture everything you need to include in the report and then delete the endpoint
- After finishing your assignment, make sure you delete the endpoint
- Save the completed notebook for the assignment into your local computer since you need to explain it in the presentation date

Assignment

- The training job you used , we started based on a pre-trained model with Abalone data
- Did that really make a difference? What if you use the data I gave you and start a new training from scratch and use LighGBM algorithm to train a new mode? (https://docs.aws.amazon.com/sagemaker/latest/dg/lightgbm.html)
- You need to write code (not console) in the notebook to do this assignment
- Use the same hyperparameters you used during the class in JS to train from scratch and compare at least the following items:
  - Training time
  - Quality of model against validation data
  - Residual values
- Write a report to explain the steps you took and what you learned out of this activity
- Submit the completed notebook and report to blackboard. The notebook should have all the cells run already and I want to see the result of each cell
- You just open the file in the presentation day and explain what happens in each cell. You do not run those cells in the presentation day. We just talk about results and reports.