ML Modeling with Amazon SageMaker
Mico Ellerich Comia
May 22, 2021

---

1. **Hyperparameter Tuning**
    - Overview
        - Tuning model to get a better model.
        - Automated model tuning is the automation of hyperparameter optimization.
            1. Creates multiple training jobs with different configurations.
        - Hyperparameters vary per algorithm.
    - Parameters vs. Hyperparameters
        - Parameters: values that change (e.g. weights and biases) once training job starts.
        - Hyperparameters: values that are constant once training job starts.

2. **Notebook 4: Automated Model Tuning with XGBoost Algorithm**
    - *sagemaker.inputs.TrainingInput()*

```
from sagemaker.inputs import TrainingInput

s3_input_training = TrainingInput(train_s3_path, content_type="text/csv")
s3_input_validation = TrainingInput(val_s3_path, content_type="text/csv")
```

        - Creates a definition for input data used by a training job
        - Stores inputs in S3
    - *(estimator).set_hyperparameters()*

```
estimator.set_hyperparameters(
    eval_metric='rmse',
    objective='reg:squarederror',
    num_round=10)
```

        - Hyperparameter values that are constant across all training jobs.
    - *hyperparameter_ranges*

```
hyperparameter_ranges = {
    'eta': tuner.ContinuousParameter(0, 1),
    'min_child_weight': tuner.ContinuousParameter(3, 7),
    'max_depth': tuner.IntegerParameter(2, 8)
}
```

- Dictionary of hyperparameter values that vary across training jobs. Ranges can be one of three types: *continuous*, *integer*, *or categorical*.
- Passed as a parameter of *tuner.HypeparameterTuner*

3. **Notebook 5: Analyzing Automated Model Tuning Results**
   - *HyperparameterTuningJobAnalytics*

```
import pandas as pd
from sagemaker import HyperparameterTuningJobAnalytics

def latest_df():
    analytics = HyperparameterTuningJobAnalytics(tuning_job_name)

    return analytics.dataframe()
```

```
latest_df().sort_values('FinalObjectiveValue', ascending=False)
```

	eta	max_depth	min_child_weight	TrainingJobName	TrainingJobStatus	FinalObjectiveValue	TrainingStartTime	TrainingEndTime
4	0.206869	5.0	6.215360	sagemaker-xgboost-210506-1941-002-dbfcf69b	Completed	6738980.0	2021-05-06 19:44:20+00:00	2021-05-06 19:45:37+00:00
3	0.090892	8.0	4.422702	sagemaker-xgboost-210506-1941-003-bd208717	Completed	6147650.0	2021-05-06 19:44:05+00:00	2021-05-06 19:45:48+00:00
5	0.099628	7.0	3.536840	sagemaker-xgboost-210506-1941-001-c92922f1	Completed	5679570.0	2021-05-06 19:44:10+00:00	2021-05-06 19:45:51+00:00
0	0.529814	2.0	3.050551	sagemaker-xgboost-210506-1941-006-dbbf39d4	InProgress	NaN	NaT	NaT
1	0.602182	7.0	3.000009	sagemaker-xgboost-210506-1941-005-b38d1498	InProgress	NaN	NaT	NaT
2	0.981230	7.0	3.008559	sagemaker-xgboost-210506-1941-004-a65667b8	InProgress	NaN	NaT	NaT

   - Gives you feedback regarding training jobs.
   - Possibly lets you know if your configuration is incorrect.
   - *SageMaker Debugger*
     - Allows you to specify rules for monitoring and detection of ML training jobs.
     - Alerts users if a rule is violated.
     - Refer to: https://docs.aws.amazon.com/sagemaker/latest/dg/train-debugger.html

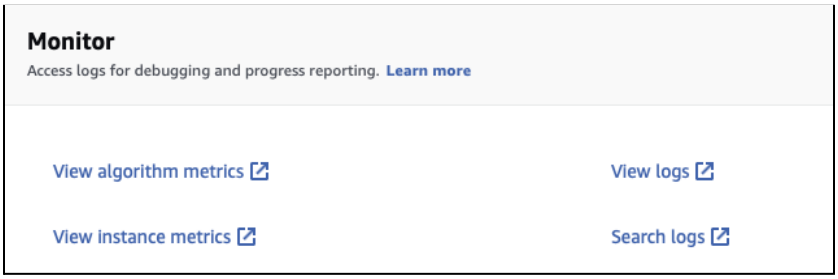
4. **Notebook 6: Challenge Lab - Automated Model Tuning with Linear Learner**
   - Hyperparameter Tuning Jobs

- Misconfiguration may occur. As such, don't panic and check logs for errors.
- In the SageMaker console, you may check the logs of hyperparameter tuning jobs.
    1. In the SageMaker console, dropdown "Training" and select "Hyperparameter tuning jobs."
    2. Select the name of your hyperparameter tuning job which you may check using *(tuner).latest_tuning_job.job_name*.

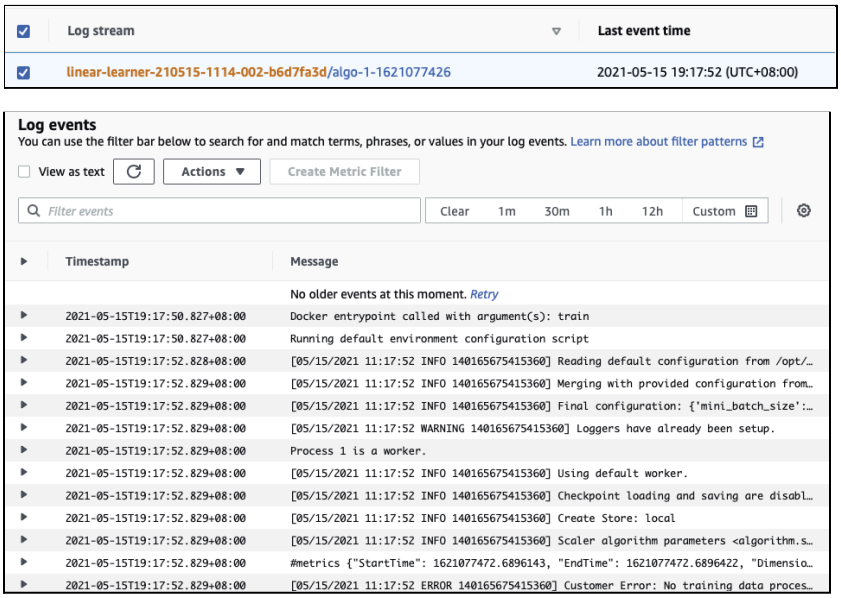
```
job_name = hyperparameter_tuner.latest_tuning_job.job_name
```

```
'HyperParameterTuningJobName': 'linear-learner-210515-0655'
```

    3. Select the training job of concern under the "Training jobs" tab. Scroll down to the "Monitor" section and click "View Logs."



    4. Check log stream to check possible cause of error.



- Other tips

        - If stuck, check for existing recipes and notebooks which you can refer to.
        - Handling blockers is an integral part of handling real-life ML problems.

5. **21-Day Challenge Project**
    - *Built-in magic commands*
        - *%store* - Store variable to allow access in other notebooks.
        - *%store -r* - Read stored value from another notebook.
        - *%time* - Time the execution of a statement or expression.
        - Other commands are documented in https://ipython.readthedocs.io/en/stable/interactive/magics.html
    - Others
        - Joblib
            - Malicious code may be injected into joblib files. Use with caution when loading third-party joblib files.
        - Notebook
            - Does not necessarily have to follow best practices (PEP 8).
            - Prioritize readability.
        - Skills
            - In addition to SageMaker, mastery of Pandas, NumPy, magic commands is recommended to excel in the field of Data Science and ML.
            - A big problem can be broken down into smaller steps.
                a. Iterative approach that allows you to catch errors early on.
                b. Makes tackling big problems manageable.