Machine Learning Modeling Day 3 Notes

SageMaker Automated Model Tuning

A way to concurrently train multiple models with different hyperparameter configurations to determine the most optimal set of hyperparameters based on a predetermined metric.

Before you start using hyperparameter tuning, you should have a well-defined machine learning problem, including the following:

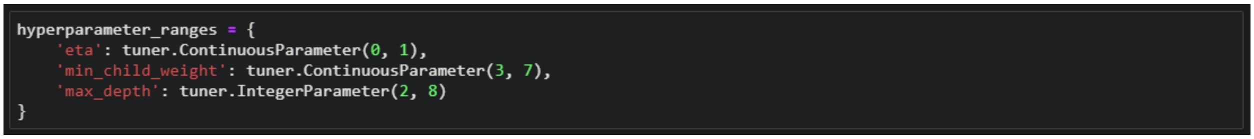
- A dataset
- An understanding of the type of algorithm you need to train
- A clear understanding of how you measure success

Parameters vs. Hyperparameters

- Parameters are set values that change once training starts.
- Hyperparameters are set values that cannot be changed once training starts and throughout the entire duration of training.

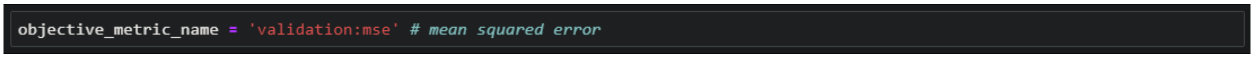
Automated Model Tuning with the XGBoost Algorithm

Hyperparameter ranges:



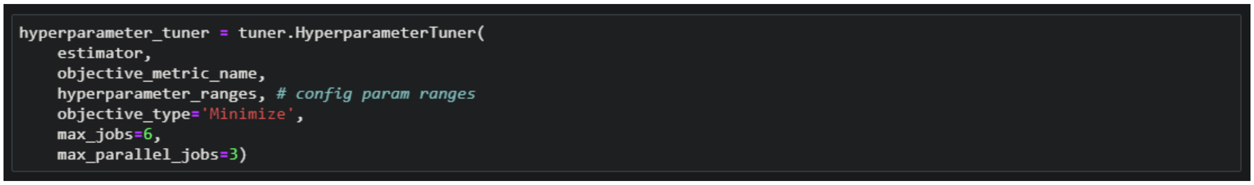
```
hyperparameter_ranges = {
    'eta': tuner.ContinuousParameter(0, 1),
    'min_child_weight': tuner.ContinuousParameter(3, 7),
    'max_depth': tuner.IntegerParameter(2, 8)
}
```

Metric:



```
objective_metric_name = 'validation:mse' # mean squared error
```

Hyperparameter tuning jobs:

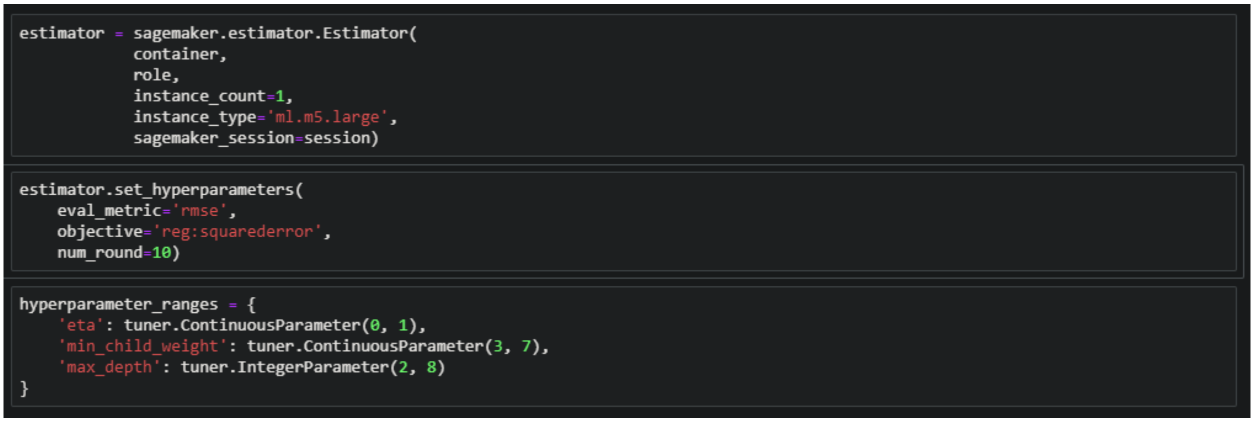


```
hyperparameter_tuner = tuner.HyperparameterTuner(
    estimator,
    objective_metric_name,
    hyperparameter_ranges, # config param ranges
    objective_type='Minimize',
    max_jobs=6,
    max_parallel_jobs=3)
```

Hyperparameter Configuration

Automated Model Tuning with the XGBoost Algorithm

Hyperparameter Configuration:



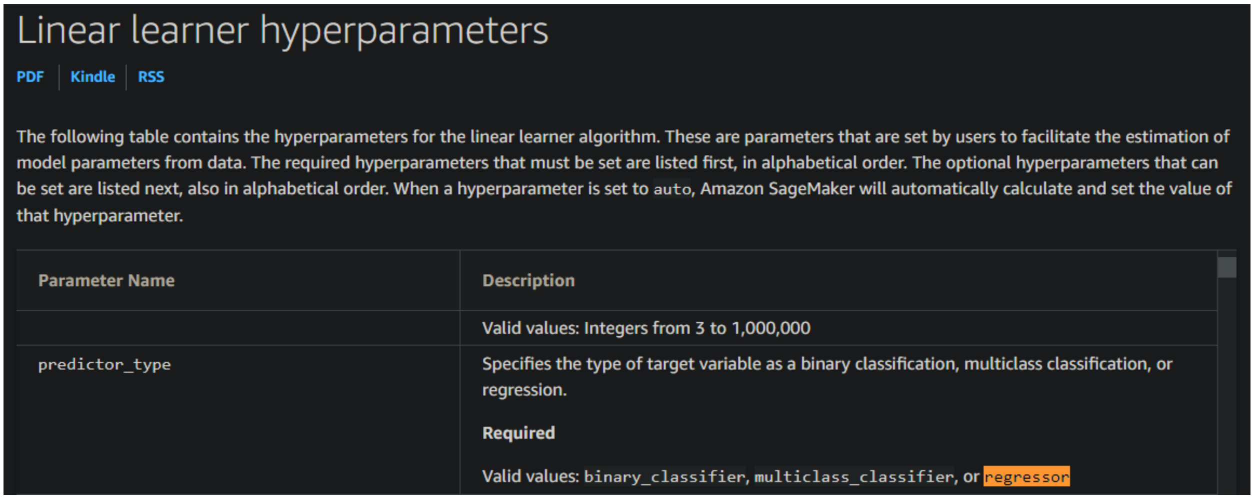
```
estimator = sagemaker.estimator.Estimator(
            container,
            role,
            instance_count=1,
            instance_type='ml.m5.large',
            sagemaker_session=session)
```

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

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

💡 the attributes for the /set_hyperparameters method are the ***required*** hyperparameters while the ones in the hyperparameter_ranges are the ***optional*** hyperparameters.

Documentation for the objective metrics for the Linear Learner algorithm

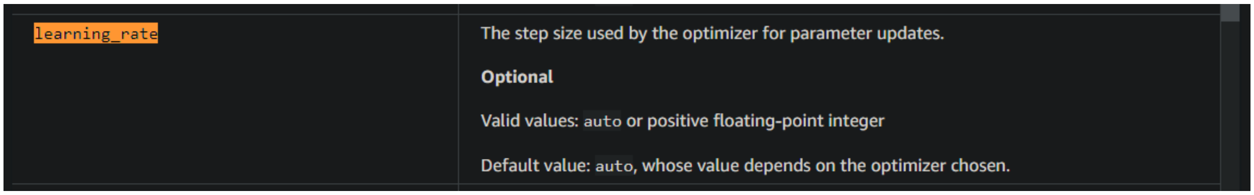


Linear learner hyperparameters

PDF | Kindle | RSS

The following table contains the hyperparameters for the linear learner algorithm. These are parameters that are set by users to facilitate the estimation of model parameters from data. The required hyperparameters that must be set are listed first, in alphabetical order. The optional hyperparameters that can be set are listed next, also in alphabetical order. When a hyperparameter is set to `auto`, Amazon SageMaker will automatically calculate and set the value of that hyperparameter.

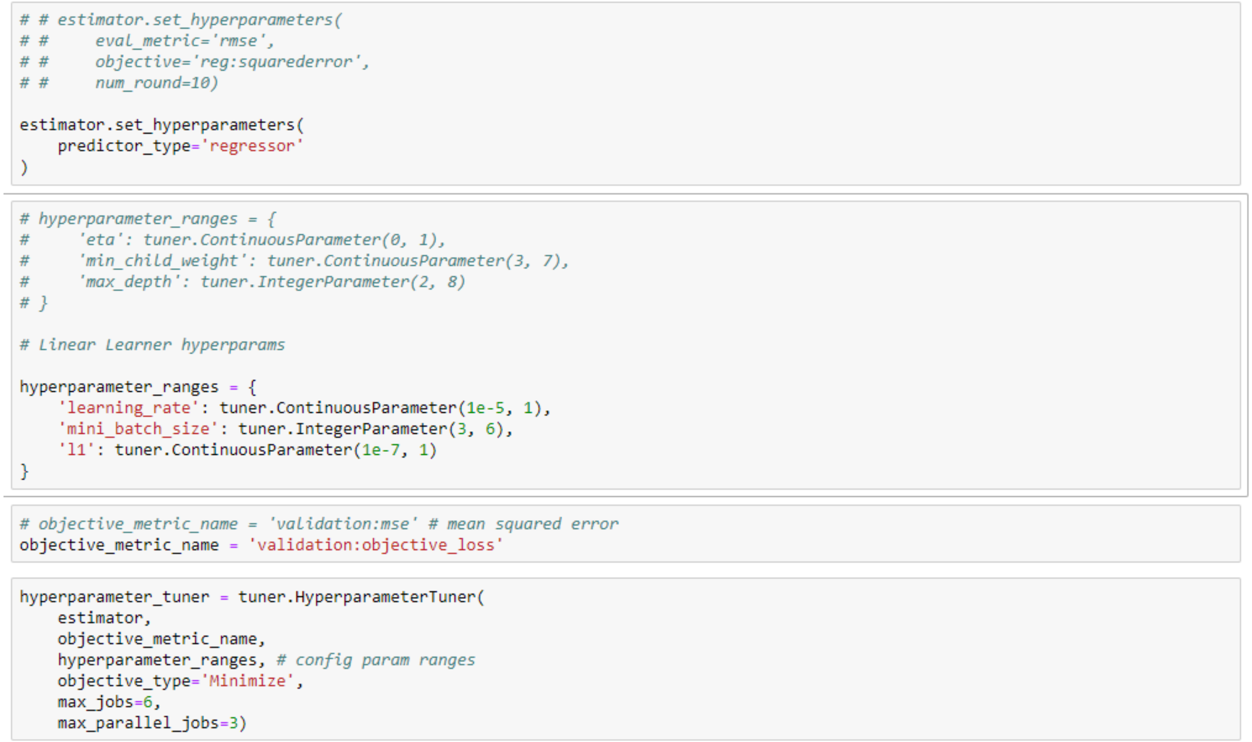
Parameter Name	Description
	Valid values: Integers from 3 to 1,000,000
`predictor_type`	Specifies the type of target variable as a binary classification, multiclass classification, or regression. **Required** Valid values: `binary_classifier`, `multiclass_classifier`, or `regressor`



`learning_rate`	The step size used by the optimizer for parameter updates. **Optional** Valid values: `auto` or positive floating-point integer Default value: `auto`, whose value depends on the optimizer chosen.

Challenge Lab - Automated Model Tuning with Linear Learner

Hyperparameter Configuration:



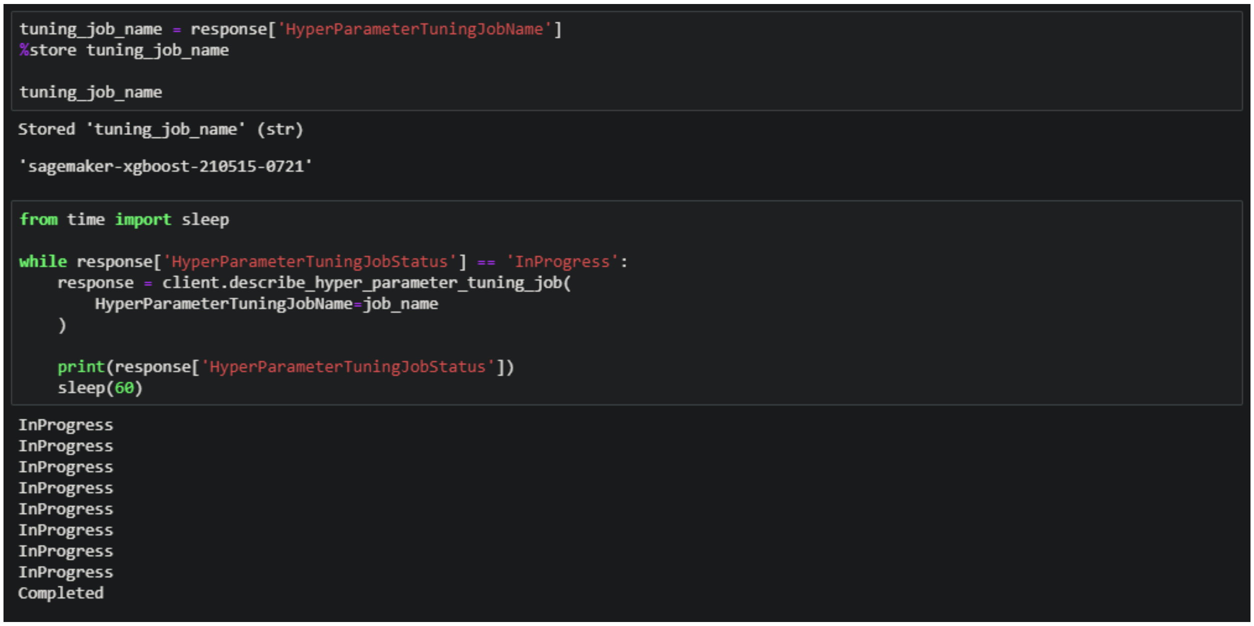
```
# # estimator.set_hyperparameters(
# #     eval_metric='rmse',
# #     objective='reg:squarederror',
# #     num_round=10)

estimator.set_hyperparameters(
    predictor_type='regressor'
)
```

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

# Linear Learner hyperparams

hyperparameter_ranges = {
    'learning_rate': tuner.ContinuousParameter(1e-5, 1),
    'mini_batch_size': tuner.IntegerParameter(3, 6),
    'l1': tuner.ContinuousParameter(1e-7, 1)
}
```

```
# objective_metric_name = 'validation:mse' # mean squared error
objective_metric_name = 'validation:objective_loss'
```

```
hyperparameter_tuner = tuner.HyperparameterTuner(
    estimator,
    objective_metric_name,
    hyperparameter_ranges, # config param ranges
    objective_type='Minimize',
    max_jobs=6,
    max_parallel_jobs=3)
```

Hyperparameter tuning and usage of Jupyter Notebook's %store magic



```
tuning_job_name = response['HyperParameterTuningJobName']
%store tuning_job_name

tuning_job_name
```

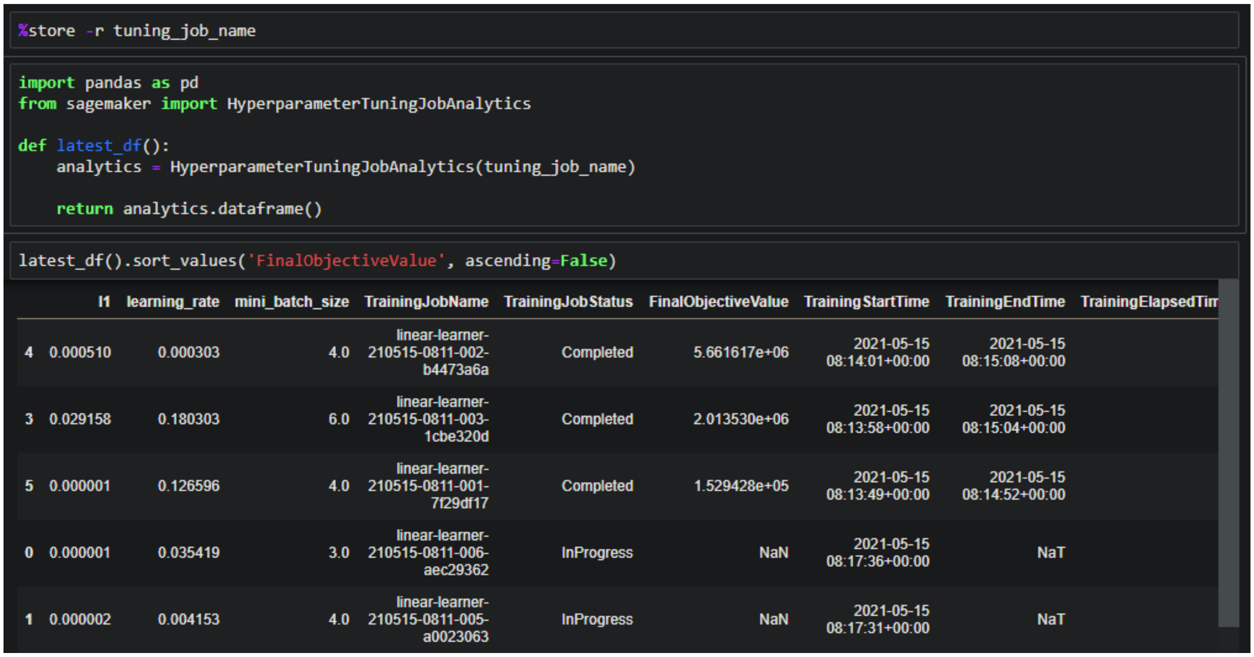
```
Stored 'tuning_job_name' (str)

'sagemaker-xgboost-210515-0721'
```

```
from time import sleep

while response['HyperParameterTuningJobStatus'] == 'InProgress':
    response = client.describe_hyper_parameter_tuning_job(
        HyperParameterTuningJobName=job_name
    )

    print(response['HyperParameterTuningJobStatus'])
    sleep(60)
```

```
InProgress
InProgress
InProgress
InProgress
InProgress
InProgress
InProgress
InProgress
Completed
```

- used for model evaluation

Analyzing Automated Model Tuning Results



```
%store -r tuning_job_name
```

```
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)
```

	l1	learning_rate	mini_batch_size	TrainingJobName	TrainingJobStatus	FinalObjectiveValue	TrainingStartTime	TrainingEndTime	TrainingElapsedTim
4	0.000510	0.000303	4.0	linear-learner-210515-0811-002-b4473a6a	Completed	5.661617e+06	2021-05-15 08:14:01+00:00	2021-05-15 08:15:08+00:00	
3	0.029158	0.180303	6.0	linear-learner-210515-0811-003-1cbe320d	Completed	2.013530e+06	2021-05-15 08:13:58+00:00	2021-05-15 08:15:04+00:00	
5	0.000001	0.126596	4.0	linear-learner-210515-0811-001-7f29df17	Completed	1.529428e+05	2021-05-15 08:13:49+00:00	2021-05-15 08:14:52+00:00	
0	0.000001	0.035419	3.0	linear-learner-210515-0811-006-aec29362	InProgress	NaN	2021-05-15 08:17:36+00:00	NaT	
1	0.000002	0.004153	4.0	linear-learner-210515-0811-005-a0023063	InProgress	NaN	2021-05-15 08:17:31+00:00	NaT	

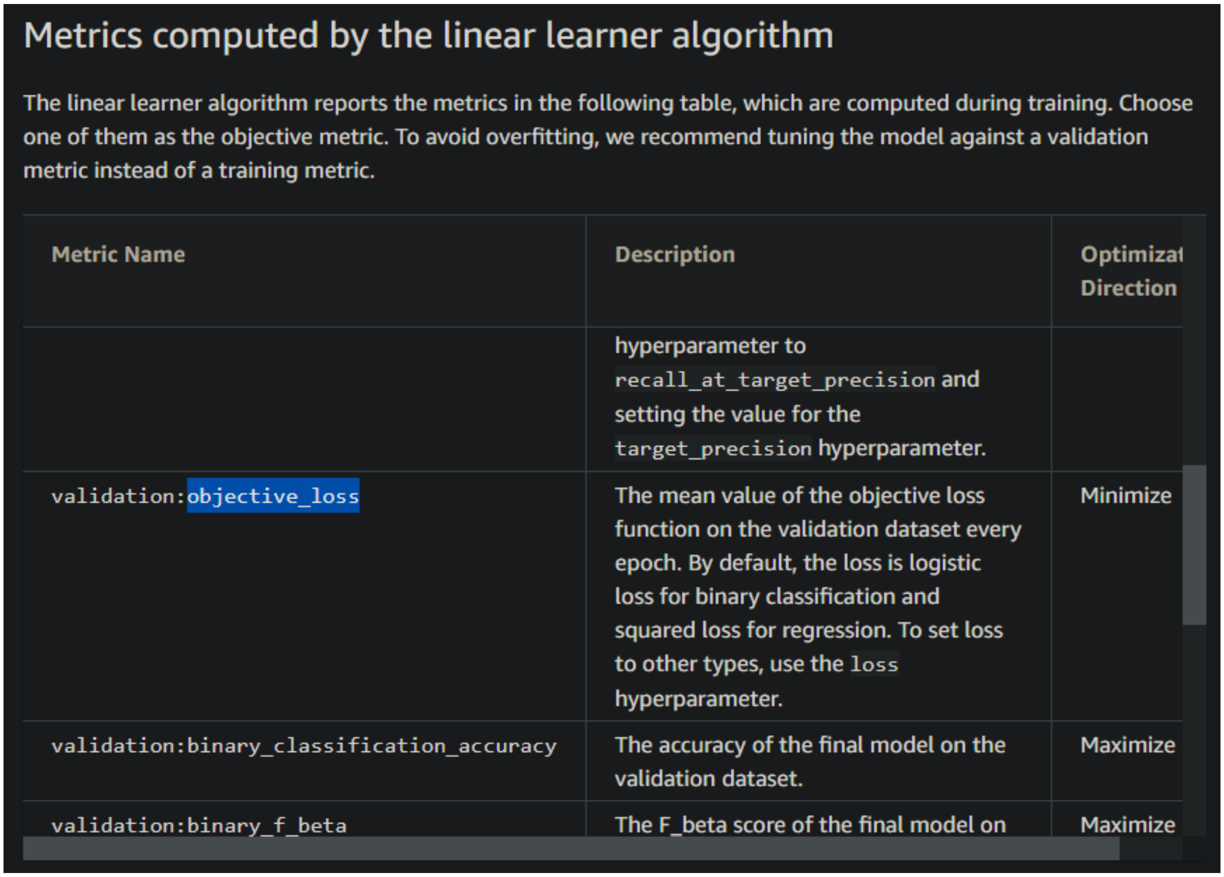
Nested Hyperparameter Tuning

- An approach to tune different algorithms or model families with different objective metrics and hyperparameter configurations.

- i.e. Solve a problem using different model families

Additional Notes

Documentation for the objective metrics for the Linear Learner algorithm



Metrics computed by the linear learner algorithm

The linear learner algorithm reports the metrics in the following table, which are computed during training. Choose one of them as the objective metric. To avoid overfitting, we recommend tuning the model against a validation metric instead of a training metric.

Metric Name	Description	Optimizat Direction
	hyperparameter to `recall_at_target_precision` and setting the value for the `target_precision` hyperparameter.	
`validation:objective_loss`	The mean value of the objective loss function on the validation dataset every epoch. By default, the loss is logistic loss for binary classification and squared loss for regression. To set loss to other types, use the `loss` hyperparameter.	Minimize
`validation:binary_classification_accuracy`	The accuracy of the final model on the validation dataset.	Maximize
`validation:binary_f_beta`	The F_beta score of the final model on	Maximize

https://docs.aws.amazon.com/sagemaker/latest/dg/linear-learner-tuning.html