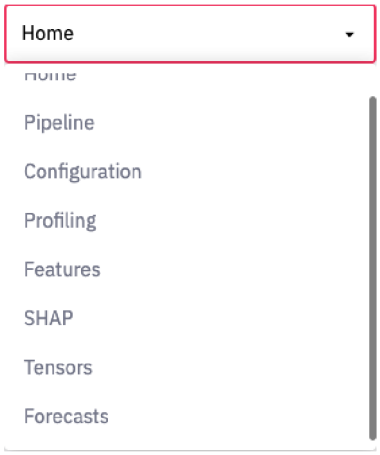
EASF Manual

First-Time User

When the app is first launched, most layouts are not available, as a complete pipeline execution is required beforehand.
For that, please go first to the *Configuration Page* and review the parameters/hyperparameters values, submit the changes if there are any.
Second, navigate to the *Pipeline Page,* enable the *Training* option (set to “True”) and click the *Run Button.* Wait until the execution is completed, and now the rest of the layouts should be available, once the data is loaded.

Layouts



Pipeline

Pipeline Flow

Configuration

Problem Formulation

Profiling

Exploratory Data Analysis

Features

Feature Engineering

SHAP

SHAP Analysis

Tensors

Tensors Analysis

Forecasts

Forecast Analysis

Hyperparameters Manual

AWS Settings

RawSourceFile: The key of the input data source.

ModelName: Name of the model definition.

EndpointName: Name of the endpoint .

TuningJobName: Name of tuning job.

TrainingJobNameDeepAR: Name of the DeepAR training job.

TrainingJobNameGluonTS: Name of the repurposed DeepAR image training job.

ProcessingJobName: Name of the preprocessing job (which includes. the data prep, dynamic features models, surrogate model, SHAP computation).

Hyperparameters Feature Enigneering

TargetVariablesList: List of target variables (independent variables).

PivotColumn: The category column (mandatory).
Note: If the dataset does not have several categories available (related time-series), the principle of multiple-related time-series is violated, thus the tool/method is likely not suitable for this use case.

TimestampColumn: The time stamp column (date).

AlgoSelection: Selecting the main algorithm used for generating the dynamic features (self-adaptive covariates). Choose between Exponential Smoothing and Neural Prophet.
Note: The exponential smoothing option has an embedded optimiser which finds the optimal seasonal smoothing coefficients for each target independently.

InterpolationMethod: Interpolation method used to deal with the missing data.
Valid options are: `nearest`, `zero`, `slinear`, `quadratic`, `cubic`, `spline`, `barycentric`, `polynomial`, `time`, `krogh`, `piecewise_polynomial`, `spline`, `pchip`, `akima`

InterpolationOrder: The interpolation order for spline and polynomial methods `(less than 5)`.

StepSize: The step size for the number of periods in a complete seasonal cycle, e.g., 4 for quarterly data or 7 for daily data with a weekly cycle. (used by the optimizer)

MinSeasonalGuess: The mimimum number of periods in a complete seasonal cycle, e.g., 4 for quarterly data or 7 for daily data with a weekly cycle. (used by the optimizer)

MaxSeasonalGuess: The maximum number of periods in a complete seasonal cycle, e.g., 4 for quarterly data or 7 for daily data with a weekly cycle. (used by the optimizer)

SeasonalSmoothingThresold: A threshold for validating the returned seasonal coefficients.(empirically set while experimenting with the specificity of the data)

TestDataSize: The size of the test fold, used by the Exponential Smoothing optimizer. (eg. if freq = 'D', this can be 7 days)

WindowsFolds: The number of folds for the walk-forward validation strategy used with the Exponential Smoothing optimizer.

RollingWindow: The rolling window size used as a smoothing technique to eliminate short-term fluctuations of the outliers score/abnormal contributions.

FeatureExpOrder: The order of the exponential transformation of the outliers score/abnormal contributions.

Contamination: The contamination threshold used to generate binary outlier labels. It is n_samples * contamination most abnormal samples.

Nneighbors: KNNImputer used in conjunction with the interpolation and the smoothing method to deal with missing data. Select the number of neighbours.

KmeansClusters: For obtaining group-level effects, time-series are clustered using TimeSeriesKMeans, using a dynamic time warping metric. Select the number of clusters.

KmeansIters: Maximum number of iterations of the k-means algorithm for a single run.

NeuralNchangepoints: The trend flexibility if primarily controlled by **NeuralNchangepoints**, which sets the number of points where the trend rate may change.

NeuralChangepointsRange: Controls the range of training data used to fit the trend. The default value of 0.8 means that no changepoints are set in the last 20 percent of training data.

NeuralYearlySeasonality, NeuralWeaklySeasonality, NeuralDailySeasonality: are about which seasonal components to be modelled. For example, if you use temperature data, you can probably select daily and yearly. Using number of passengers using the subway would more likely have a weekly seasonality for example. Setting these seasonalities at the default `auto` mode, lets NeuralProphet decide which of them to include depending on how much data available. For example, the yearly seasonality will not be considered if less than two years data available. In the same manner, the weekly seasonality will not be considered if less than two weeks available etc... However, if the user if certain that the series does not include yearly, weekly or daily seasonality, and thus the model should not be distorted by such components, they can explicitly turn them off by setting the respective components to `False`.

NeuralNforecasts: The size of the forecast horizon. The default value of 1 means that the model forecasts one step into the future.

NeuralNlags: NeuralNlags defines whether the AR-Net is enabled (if **NeuralNlags** > 0) or not. The value for **NeuralNlags** is usually recommended to be greater than **NeuralNforecasts**, if possible since it is preferable for the FFNNs to encounter at least **NeuralNforecasts** length of the past in order to predict **NeuralNforecasts** into the future. Thus, **NeuralNlags** determine how far into the past the auto-regressive dependencies should be considered. This could be a value chosen based on either domain expertise or an empirical analysis.

Hyperparameters DeepAR

EpochsDeepAR: The maximum number of passes over the training data. The optimal value depends on your data size and learning rate. Typical values range from 10 to 1000.

PredictionLength: The number of time-steps that the model is trained to predict, also called the forecast horizon. The trained model always generates forecasts with this length. It can't generate longer forecasts. The **PredictionLength** is fixed when a model is trained and it cannot be changed later.

ContextLenght: The number of time-points that the model gets to see before making the prediction. The value for this parameter should be about the same as the **PredictionLength**. The model also receives lagged inputs from the target, so **ContextLenght** can be much smaller than typical seasonalities. For example, a daily time series can have yearly seasonality. The model automatically includes a lag of one year, so the context length can be shorter than a year. The lag values that the model picks depend on the frequency of the time series. For example, lag values for daily frequency are previous week, 2 weeks, 3 weeks, 4 weeks, and year.

NumLayersDeepAR: The number of hidden layers in the RNN. Typical values range from 1 to 4.

DropoutRateDeepAR: The dropout rate to use during training. The model uses zoneout regularization. For each iteration, a random subset of hidden neurons are not updated. Typical values are less than 0.2.

LearningRateDeepAR: The learning rate used in training. Typical values range from 1e-4 to 1e-1.

NumCellsDeepAR: The number of cells to use in each hidden layer of the RNN. Typical values range from 30 to 100.

TimeseriesFreq: The granularity of the time series in the dataset. Use **TimeseriesFreq** to select appropriate date features and lags. The model supports the following basic frequencies. It also supports multiples of these basic frequencies. For example, `5min` specifies a frequency of 5 minutes.

M: monthly
W: weekly
D: daily
H: hourly
min: every minute

TrainInstanceTypeDeepAR: Instance Type.

TrainInstanceCount: Count of instances.

RMSEthresholdDeepAR: The threshold set for the test RMSE when promoting the model.

Save_intervalGluonTS: The steps interval for saving the tensors.
Note**:** Should not be smaller than 100 as it would significantly hinder the performance without increased benefits.

Hyperparameters Optimization

HPOEnabled: Enable the hyper parameter optimisation.

MaxJobsHPODeepAR: Maximum number of jobs to be run.

MaxParallelJobsHPODeepAR: How many parallel jobs to be run.

ObjectiveMetricDeepAR: Valid options are:

test:RMSE The root mean square error between the forecast and the actual target computed on the test set.
test:mean_wQuantileLoss The average overall quantile losses computed on the test set.
train:final_loss The training negative log-likelihood loss averaged over the last training epoch for the model.

LearningRateDeepAR_min: MinValue: 1e-5
LearningRateDeepAR_max: MaxValue: 1e-1
EpochsDeepAR_min: MinValue: 1
EpochsDeepAR_max: MaxValue**:** 1000
NumCellsDeepAR_min: MinValue**:** 30

NumCellsDeepAR_max: MaxValue**:** 200
NumLayersDeepAR_min: MinValue**:** 1
NumLayersDeepAR_max: MaxValue**:** 8
ContextLenght_min: MinValue**:** 1
ContextLenght_max: MaxValue**:** 200

Hyperparameters Surrogate Model

ShapInteractionFlag: Enable the calculation of SHAP interaction values.

XgboostEstimators: Number of gradient boosted trees. Equivalent to number of boosting rounds.

XgboostDepth: Maximum depth of a tree. Increasing this value will make the model more complex and more likely to overfit.

XgboostShapSamples: The maximum number of samples to use from the passed background data. If data has more than max_samples then shap.utils.sample is used to subsample the dataset. The number of samples coming out of the masker (to be integrated over) matches the number of samples in the background dataset. This means larger background dataset cause longer runtimes. Normally about 1, 10, 100, or 1000 background samples are reasonable choices.

XgboostEta: Step size shrinkage used in update to prevents overfitting. After each boosting step, we can directly get the weights of new features, and `eta` shrinks the feature weights to make the boosting process more conservative.

XgboostNjobs: Number of parallel threads used to run xgboost.

XgboostGamma: Minimum loss reduction required to make a further partition on a leaf node of the tree. The larger `gamma` is, the more conservative the algorithm will be.