Amazon Forecast

Developer Guide

Amazon Forecast: Developer Guide

Table of Contents

What Is Amazon Forecast?

Amazon Forecast is a fully managed service for time-series forecasting. By providing Amazon Forecast with historical time-series data, you can predict future points in the series. Time-series forecasting is useful in multiple domains, including retail, financial planning, supply chain, and healthcare. You can also use Amazon Forecast to forecast operational metrics for inventory management, and workforce and resource planning and management.

For example, you can use Amazon Forecast to forecast the following:

- Retail product demand, such as the demand for products selling on a website or at a particular store or location
- Supply chain demand including the quantity of raw goods, services, or other inputs needed by manufacturing
- Resource requirements, such as the number of call center agents, contract workers, IT staff, and energy needed to meet demand
- Operational metrics, such as web traffic to servers, AWS usage, or IoT sensor usage
- Business metrics, such as cash flow, sales, profits, and expenses on a per-region or per-service basis

Amazon Forecast greatly simplifies building machine learning models. In addition to providing a set of predefined algorithms, Forecast provides an *AutoML* option for model training. AutoML automates complex machine learning tasks, such as algorithm selection, hyperparameter tuning, iterative modeling, and model assessment. Developers with no machine learning expertise can use the Amazon Forecast APIs, AWS Command Line Interface (AWS CLI), or Amazon Forecast console to import training data into one or more Amazon Forecast datasets, train predictors, and generate forecasts.

Amazon Forecast provides the following additional advantages:

- Accuracy – Amazon Forecast uses deep neural net and traditional statistical methods for forecasting. When you have many related time series, forecasts made using the Amazon Forecast deep learning algorithms, such as DeepAR+ (p. 56), tend to be more accurate than forecasts made with traditional methods, such as exponential smoothing.
- Usability – You can use theAmazon Forecast console to look up and visualize forecasts for any time series at different granularities. You can also see metrics for the accuracy of your forecasts.

For more information on Amazon Forecast, including use cases and underlying service principles, see Time Series Forecasting Principles with Amazon Forecast.

Are You a First-Time User of Amazon Forecast?

If you are a first-time user of Amazon Forecast, we recommend that you read the following:

How Amazon Forecast Works

When creating forecasting projects in Amazon Forecast, you work with the following resources:

- **Datasets and Dataset Groups (p. 2)** – *Datasets* are collections of your input data. Dataset groups are collections of datasets that contain complimentary information. Forecast algorithms use your dataset groups to train custom forecasting models, called predictors.
- **Predictors (p. 12)** – *Predictors* are custom models trained on your data. You can train a predictor by choosing a prebuilt algorithm,or by choosing the AutoML option to have Amazon Forecast pick the best algorithm for you.
- **Forecasts (p. 13)** – You can generate forecasts for your time-series data, query them using the QueryForecast API, or visualize them in the console.

Topics

Datasets and Dataset Groups

Datasets contain the data used to train a predictor (p. 12). You create one or more Amazon Forecast datasets and import your training data into them. A *dataset group* is a collection of complimentary datasets that detail a set of changing parameters over a series of time. After creating a dataset group, you use it to train a predictor.

Each dataset group can have up to three datasets, one of each dataset (p. 3) type: target time series, related time series, and item metadata.

To create and manage Forecast datasets and dataset groups, you can use the Forecast console, AWS Command Line Interface (AWS CLI), or AWS SDK.

For example Forecast datasets, see the Amazon Forecast Sample GitHub repository.

Topics

Datasets

To create and manage Forecast datasets, you can use the Forecast APIs, including the CreateDataset (p. 113) and DescribeDataset (p. 147) operations. For a complete list of Forecast APIs, see API Reference (p. 111).

When creating a dataset, you provide information, such as the following:

- The frequency/interval at which you recorded your data. For example, you might aggregate and record retail item sales every week. In the Getting Started (p. 19) exercise, you use the average electricity used per hour.
- The prediction format (the *domain*) and dataset type (within the domain). A dataset domain specifies which type of forecast you'd like to perform, while a dataset type helps you organize your training data into Forecast-friendly categories.
- The dataset *schema*. A schema maps the column headers of your dataset. For instance, when monitoring demand, you might have collected hourly data on the sales of an item at multiple stores. In this case, your schema would define the order, from left to right, in which timestamp, location, and hourly sales appear in your training data file. Schemas also define each column's data type, such as `string` or `integer`.

Each column in your Forecast dataset represents either a forecast *dimension* or *feature*. Forecast dimensions describe the aspects of your data that do not change over time, such a `store` or `location`. Forecast features include any parameters in your data that vary across time, such as `price` or `promotion`. Some dimensions, like `timestamp` or `itemId`, are required in target time series and related time series datasets.

Dataset Domains and Dataset Types

When you create a Forecast dataset, you choose a domain and a dataset type. Forecast provides domains for a number of use cases, such as forecasting retail demand or web traffic. You can also create a custom domain. For a complete list of Forecast domains, see Predefined Dataset Domains and Dataset Types (p. 46).

Within each domain, Forecast users can specify the following types of datasets:

- Target time series dataset (required) – Use this dataset type when your training data is a time series *and* it includes the field that you want to generate a forecast for. This field is called the *target field*.
- Related time series dataset (optional) – Choose this dataset type when your training data is a time series, but it *doesn't* include the target field. For instance, if you're forecasting item demand, a related time series dataset might have `price` as a field, but not `demand`.
- Item metadata dataset (optional) – Choose this dataset type when your training data *isn't* time-series data, but includes metadata information about the items in the target time series or related time series datasets. For instance, if you're forecasting item demand, an item metadata dataset might `color` or `brand` as dimensions. Forecast only considers the data provided by an item metadata dataset type when you use the DeepAR+ (p. 56) algorithm.

Depending on the information in your training data and what you want to forecast, you might create more than one dataset.

For example, suppose that you want to generate a forecast for the demand of retail items, such as shoes and socks. You might create the following datasets in the RETAIL domain:

- Target time series dataset – Includes the historical time-series demand data for the retail items (`item_id`, `timestamp`, and the target field `demand`). Because it designates the target field that you want to forecast, you must have at least one target time series dataset in a dataset group.

  You can also add up to ten other dimensions to a target time series dataset. If you include only a target time series dataset in your dataset group, you can create forecasts at either the item level or the forecast dimension level of granularity only. For more information, see CreatePredictor (p. 129).
- Related time series dataset – Includes historical time-series data other than the target field, such as `price` or `revenue`. Because related time series data must be mappable to target time series data, each related time series dataset must contain the same identifying fields. In the RETAIL domain, these would be `item_id` and `timestamp`.

A related time series dataset might contain data that refines the forecasts made off of your target time series dataset. For example, you might include `price` data in your related time series dataset on the future dates that you want to generate a forecast for. This way, Forecast can make predictions with an additional dimension of context. For more information, see Using Related Time Series Datasets (p. 7).

- Item metadata dataset – Includes metadata for the retail items. Examples of metadata include `brand`, `category`,`color`, and `genre`.

Example Dataset with a Forecast Dimension

Continuing with the preceding example, imagine that you want to forecast the demand for shoes and socks based on a store's previous sales. In the following target time series dataset, `store` is a time-series forecast dimension, while `demand` is the target field. Socks are sold in two store locations (NYC and SFO), and shoes are sold only in ORD.

The first three rows of this table contain the first available sales data for the NYC, SFO, and ORD stores. The last three rows contain the last recorded sales data for each store. The `...` row represents all of the item sales data recorded between the first and last entries.

timestamp	item_id	store	demand
2019-01-01	socks	NYC	25
2019-01-05	socks	SFO	45
2019-02-01	shoes	ORD	10
...			
2019-06-01	socks	NYC	100
2019-06-05	socks	SFO	5
2019-07-01	shoes	ORD	50

Dataset Schema

Each dataset requires a schema, a user-provided JSON mapping of the fields in your training data. This is where you list both the required and optional dimensions and features that you want to include in your dataset.

Some domains have optional dimensions that we recommend including. Optional dimensions are listed in the descriptions of each domain later in this guide. For an example, see RETAIL Domain (p. 47). All optional dimensions take the data type `string`.

A schema is required for every dataset. The following is the accompanying schema for the example target time series dataset above.

```
{
    "attributes": [
        {
            "AttributeName": "timestamp",
            "AttributeType": "timestamp"
        },
        {
            "AttributeName": "item_id",
```

```
            "AttributeType": "string"
        },
        {
            "AttributeName": "store",
            "AttributeType": "string"
        },
        {
            "AttributeName": "demand",
            "AttributeType": "float"
        }
    ]
}
```

When you upload your training data to the dataset that uses this schema, Forecast assumes that the `timestamp` field is column 1, the `item_id` field is column 2, the `store` field is column 3, and the `demand` field, the *target* field, is column 4.

For the related time series dataset type, all related features must have a float or integer attribute type. For the item metadata dataset type, all features must have a string attribute type. For more information, see SchemaAttribute (p. 231).

> **Note**
> An `attributeName` and `attributeType` pair is required for every column in the dataset. Forecast reserves a number of names that can't be used as the name of a schema attribute. For the list of reserved names, see Reserved Field Names (p. 91).

Dataset Groups

A *dataset group* is a collection of one to three complimentary datasets, one of each dataset type. You import datasets to a dataset group, then use the dataset group to train a predictor.

Forecast includes the following operations to create dataset groups and add datasets to them:

- CreateDatasetGroup (p. 117)
- UpdateDatasetGroup (p. 188)

Resolving Conflicts in Data Collection Frequency

Forecast can import data that isn't aligned with the collection frequency specified in the CreateDataset (p. 113) operation. For example, you can import data for which the collection frequency is hourly and some of the data isn't timestamped at the top of the hour (02:20, 02:45). Forecast aggregates the data to match the aligned value. The following tables show an example aggregation.

Pre-transformation

Time	Data	At Top of the Hour
2018-03-03 01:00:00	100	Yes
2018-03-03 02:20:00	50	No
2018-03-03 02:45:00	20	No
2018-03-03 04:00:00	120	Yes

Post-transformation

Time	Data	Notes
2018-03-03 01:00:00	100	
2018-03-03 02:00:00	70	Sum of the values between 02:00:00-02:59:59 (50 + 20)
2018-03-03 03:00:00	Empty	No values between 03:00:00-03:59:59
2018-03-03 04:00:00	120	

Time Boundaries

Time Boundaries

The following table lists the time alignment boundaries Forecast uses when aggregating data.

Frequency	Boundary
Year	First day of the year (January 1)
Month	First day of the month
Week	Most recent Monday
Hour	Last top of the hour (09:00:00, 13:00:00)
Minute	Last top of the minute (45:00, 06:00)

The following figure shows how Forecast transforms data to fit the weekly boundary:
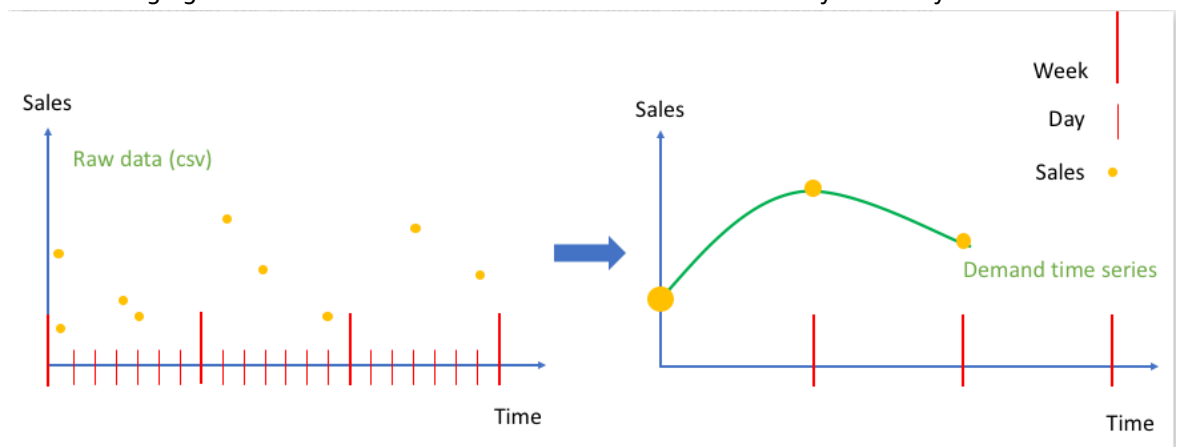
Data Aggregation Guidelines

Forecast doesn't assume that your data is from any specific time zone. However, it makes the following assumptions when aggregating time series data:

- All data is from the same time zone.
- All forecasts are in the same time zone as the data in the dataset.

- If you specify the the section called "SupplementaryFeature" (p. 234) holiday feature in the the section called "InputDataConfig" (p. 219) parameter for the the section called "CreatePredictor" (p. 129) operation, the input data is from the same country.

Using Related Time Series Datasets

A related time series dataset includes time-series data that isn't included in a target time series dataset and might improve the accuracy of your predictor.

For example, in the demand forecasting domain, a target time series dataset would contain `timestamp` and `item_id` dimensions, while a complimentary related time series dataset also includes the following supplementary features: `item price`, `promotion`, and `weather`.

A related time series dataset can contain up to 10 forecast dimensions (the same ones in your target time series dataset) and up to 13 related time-series features.

You can use a related time series dataset only when training a predictor with the DeepAR+ (p. 56) and Prophet (p. 66) algorithms. The NPTS (p. 63) algorithm and the R open-source algorithms (ARIMA (p. 55) and ETS (p. 62)) don't take data in a related time series dataset into consideration.

Related Time Series Dataset Validation

A related time series dataset has the following restrictions:

- It can't include the target value from the target time series.
- It must include `item_id` and `timestamp` dimensions, and at least one related feature (such as `price`).
- Related time series feature data must be of the `int` or `float` datatypes.
- The frequency at which data is recorded in the related time series dataset must match the interval at which you want to generate forecasts (the forecasting *granularity*).

  For example, if you want to generate forecasts at a weekly granularity, the frequency at which data is recorded in the related time series must also be weekly, even if the frequency at which data is recorded in the target time series is daily.
- The data for each item in the related time series dataset must start on or before the beginning `timestamp` of the corresponding `item_id` in the target time series dataset.

  For example, if the target time series data for `socks` starts at 2019-01-01 and the target time series data for `shoes` starts at 2019-02-01, the related time series data for `socks` must begin on or before 2019-01-01 and the data for `shoes` must begin on or before 2019-02-01.
- The last timestamp for every item in the related time series dataset must be on or after the last timestamp in the target time series *plus* the user-designated forecast window (called the *forecast horizon*).

  In the example related time series file below, the `timestamp` data for both socks and shoes must end on or after 2019-07-01 (the last recorded timestamp) *plus* the forecast horizon. If data frequency in the target time series is daily and the forecast horizon is 10 days, daily data points must be provided in the related time series file until 2019-07-11.
- The Forecast dimensions provided in the related time series dataset must be either equal to or a subset of the dimensions designated in the target time series dataset.

  Important
  Forecast doesn't support aggregations or filling missing values for related time series datasets as it does for target time series datasets.

Example: Related Time Series File

The following table shows a correctly configured related time series dataset file. For this example, assume the following:

- The last data point was recorded in the target time series dataset on 2019-07-01.
- The forecast horizon is 10 days.
- The forecast granularity is daily (D).

This means that the user had to include data points up until 2019-07-11.

A "…" row indicates all of the data points in between the previous and succeeding rows.

timestamp	item_id	store	price
2019-01-01	socks	NYC	10
2019-01-02	socks	NYC	10
2019-01-03	socks	NYC	15
...			
2019-06-01	socks	NYC	10
...			
2019-07-01	socks	NYC	10
...			
2019-07-11	socks	NYC	20
2019-01-05	socks	SFO	45
...			
2019-06-05	socks	SFO	10
...			
2019-07-01	socks	SFO	10
...			
2019-07-11	socks	SFO	30
2019-02-01	shoes	ORD	50
...			
2019-07-01	shoes	ORD	75
...			
2019-07-11	shoes	ORD	60

Example: Forecasting Granularity

The following table shows compatible data recording frequencies for target time series and related time series to forecast at a weekly granularity. Because data in a related time series dataset can't be aggregated, Forecast accepts only a related time series data frequency that is the same as the chosen forecasting granularity.

Target Input Data Frequency	Related Time Series Frequency	Forecasting Granularity	Supported by Forecast?
Daily	Weekly	Weekly	Yes
Weekly	Weekly	Weekly	Yes
N/A	Weekly	Weekly	Yes
Daily	Daily	Weekly	No

Using Item Metadata Datasets

An *item metadata dataset* contains categorical data that provides valuable context for the items in a target time-series dataset when you train a predictor with the DeepAR+ algorithm (p. 56). Unlike related time-series datasets, item metadata datasets provide information that is static. That is, the data values remain constant over time, like an item's color or brand. Item metadata datasets are optional additions to your dataset groups, and are taken into account only when you train a predictor with the DeepAR+ algorithm (p. 56). You can use an item metadata only if every item in your target time-series dataset is present in the corresponding item metadata dataset.

Item metadata might include the brand, color, model, category, place of origin, or other supplemental feature of a particular item. For example, an item metadata dataset might provide context for some of the demand data found in a target time-series dataset that represents the sales of black Amazon e-readers with 32 GB of storage. Because these characteristics don't change from day-to-day or hour-to-hour, they belong in an item metadata dataset.

Item metadata is useful for discovering and tracking descriptive patterns across your time-series data. If you include an item metadata dataset in your dataset group, Forecast can train the model to make more accurate predictions based on similarities across items. For example, you might find that virtual assistant products made by Amazon are more likely to sell out than those created by other companies, and then plan your supply chain accordingly.

Item metadata is especially useful in coldstart forecasting scenarios, in which you have little direct historical data with which to make predictions, but do have historical data on items with similar metadata attributes. When you provide context for the little data that you have, your Forecast predictor can make useful, nonobvious inferences about the items in your data that increase prediction accuracy.

Each row in an item metadata dataset can contain up to 10 metadata fields, one of which must be an identification field to match the metadata to an item in the target time series. As with all dataset types, the values of each field are designated by a dataset schema.

Example: Item Metadata File and Schema

The following table shows a section of a correctly configured item metadata dataset file that describes Amazon e-readers. For this example, assume that the header row represents the dataset's schema, and that each listed item is in a corresponding target time-series dataset. Notice that `waterproof`, a binary feature, is represented as 0 for no and 1 for yes.

item_id	brand	model	color	waterproof
1	amazon	paperwhite	black	1
2	amazon	paperwhite	blue	1
3	amazon	base_model	black	0
4	amazon	base_model	white	0
...				

The following is the same information represented in CSV format.

```
1,amazon,paperwhite,black,1
2,amazon,paperwhite,blue,1
3,amazon,base_model,black,0
4,amazon,base_model,white,0
...
```

The following is the schema for this example dataset.

```
{
    "attributes": [
        {
            "AttributeName": "item_id",
            "AttributeType": "string"
        },
        {
            "AttributeName": "brand",
            "AttributeType": "string"
        },
        {
            "AttributeName": "model",
            "AttributeType": "string"
        },
        {
            "AttributeName": "color",
            "AttributeType": "string"
        },
        {
            "AttributeName": "waterproof",
            "AttributeType": "integer"
        }
    ]
}
```

See Also

For an in-depth walkthrough on using item metadata datasets, see Incorporating Item Metadata Datasets into Your Predictor in the Amazon Forecast Samples GitHub Repository.

Dataset Guidelines for Forecast

Consult to the following guidelines if Amazon Forecast fails to import your dataset, or if your dataset doesn't function as expected.

Timestamp Format

For Year (`Y`), Month (`M`), Week (`W`), and Day (`D`) collection frequencies, Forecast supports the `yyyy-MM-dd` timestamp format (for example, `2019-08-21`) and, optionally, the `HH:mm:ss` format (for example, `2019-08-21 15:00:00`).

For Hour (`H`) and Minute (`M`) frequencies, Forecast supports only the `yyyy-MM-dd HH:mm:ss` format (for example `2019-08-21 15:00:00`).

Guideline: Change the timestamp format for the collection frequency of your dataset to the supported format.

Amazon S3 File or Bucket

When you import a dataset, you can specify either the path to the CSV file in your Amazon Simple Storage Service (Amazon S3) bucket that contains your data or the name of the S3 bucket that contains your data If you specify a CSV file, Forecast imports just that file. If you specify an S3 bucket, Forecast imports all of the CSV files in the bucket up to 10,000 files. If you import multiple files by specifying a bucket name, all CSV files must conform to the specified schema.

Guideline: Specify a CSV file or an S3 bucket using the following syntax:

```
s3://bucket-name/example-object.csv
```

```
s3://bucket-name/prefix/
```

```
s3://bucket-name
```

Dataset Updates

Because dataset import jobs are not aggregated, your most recent dataset import is the one that is used when training a predictor or generating a forecast.

Guideline: Make sure that your most recent dataset import contains all of the data you want to model off of, and not just the new data collected since the previous import.

Attribute Order

The order of attributes specified in the schema definition must match the column order in the CSV file that you are importing. For example, if you defined `timestamp` as the first attribute, then `timestamp` must also be the first column in the input CSV file.

Guideline: Verify that the columns in the CSV file are in the same order as the schema attributes that you created.

Dataset Header

A dataset header in your input CSV file may cause a validation error. We recommend omitting a header.

Guideline: Delete the dataset header and try the import again.

Dataset Status

Before you can import training data with the operation, the `Status`of the dataset must be `ACTIVE`.

Guideline: Use the operation to get the dataset's status. If the creation or update of the dataset failed, check the formatting of your dataset file and attempt to create it again.

File Format and Delimiter

Forecast supports only the comma-separated values (CSV) file format. You can't separate values using tabs, spaces, colons, or any other characters.

Guideline: Convert your dataset to CSV format (using only commas as your delimiter) and try importing the file again.

File Name

File names must contain at least one alphabetic character. Files with names that are only numeric can't be imported.

Guideline: Rename your CSV file to include at least one alphabetic character and try importing the file again.

Predictors

A predictor is an Amazon Forecast trained model used for making forecasts based on time-series data. During training, Amazon Forecast generates accuracy metrics that you use to evaluate the predictor and decide whether to use the predictor to generate a forecast.

Topics

Creating Predictors

Amazon Forecast trains forecasting models called predictors. To create a predictor, you use the CreatePredictor (p. 129) operation.

To create a predictor, you provide the following:

- A dataset group – Provides data for training the predictor. For more information, see Datasets (p. 2).
- A featurization configuration – Specifies the forecast frequency and provides information for transforming the data before model training. Data is transformed to make it more compatible with the training algorithm.
- A forecast horizon – The number of time-steps to make. The forecast horizon is also called the prediction length.
- Evaluation parameters – How to split a dataset into training and test datasets.
- One of the following:
  - An algorithm – The algorithm is used to train a model and specifies default values for hyperparameter optimization (only DeepAR+), evaluation parameters, and training parameters. By specifying an algorithm, you also can provide overrides for these parameter values.
  - Perform AutoML – Amazon Forecast provides a set of predefined algorithms. If you don't know which algorithm to choose, use the `PerformAutoML` option. This option tells Amazon Forecast to evaluate all algorithms and choose the best algorithm based on your datasets. With this option, model training can take longer, but you don't need to worry about choosing the right algorithm and parameters. AutoML optimizes the average of the weighted P10, P50 and P90 quantile losses, and returns the algorithm with the lowest value.

For more information on algorithms, see Choosing an Amazon Forecast Algorithm (p. 55).

Predictor Evaluation

After you create a predictor, you can evaluate the accuracy of the forecast it generates by running the GetAccuracyMetrics (p. 170) operation.

Evaluation Parameters

The evaluation parameters define how to split a dataset into training and test datasets for backtest window evaluations, as well as the number of backtest iterations to perform. These parameters have default values that can be overridden in the CreatePredictor (p. 129) request.

The evaluation parameters consist of the `NumberOfBacktestWindows` and the `BackTestWindowOffset` parameters.

`NumberOfBacktestWindows` specifies the number of times to split the input data. The range is 1 through 5.

`BackTestWindowOffset` defines the point from the end of the dataset where the data is split for model training and testing (evaluation). The value is specified as the number of data points. `BackTestWindowOffset` must be greater than or equal to the forecast horizon and less than half of the target time series dataset length. This parameter can be used to mimic a past virtual forecast start date.

For more information, see Evaluating Predictor Accuracy (p. 67).

How It Works: Next Topic

Forecasts (p. 13)

Forecasts

After creating an Amazon Forecast predictor, you call the CreateForecast (p. 123) operation to create a forecast. During forecast creation, Amazon Forecast trains a model on the entire dataset before hosting the model and doing inference. This operation creates a forecast for every item (`item_id`) in the dataset group that was used to train the predictor. After a forecast is created, you can query the forecast or export it to your Amazon Simple Storage Service (Amazon S3) bucket.

By default, the forecast frequency is the data collection frequency that you specified when you created the dataset with the CreateDataset (p. 113) operation. You can optionally specify an interval that is greater, but not lesser, than the specified frequency for the dataset. Then, the operation aggregates the forecast data and returns the results. For example, suppose that your data collection frequency was every day. You can then get a daily or monthly forecast, but not an hourly forecast.

You query a forecast using the QueryForecast (p. 190) operation. By default, the complete range of the forecast is returned. You can request a specific date range within the complete forecast.

When you query a forecast you must specify filtering criteria. A filter is a key-value pair. The key is one of the schema attribute names (including forecast dimensions) from one of the datasets used to create the forecast. The *value* is a valid values for the specified key. You can specify multiple key-value pairs. The returned forecast will only contain items that satisfy all the criteria.

To export the forecast, you can call the CreateForecastExportJob (p. 126) operation. This operation copies the forecast to your Amazon S3 bucket as a CSV file. Optionally, you can specify an AWS Key Management Service key to encrypt the data before it is written to the bucket.

How It Works: Next Topic

Getting Started (p. 19)

Setting Up

Before using Amazon Forecast to evaluate or forecast time-series data, create an AWS account, configure access permissions, and set up the AWS Command Line Interface (AWS CLI).

Topics
- Sign Up for AWS (p. 14)
- Set Up the AWS CLI (p. 14)
- Set Up Permissions for Amazon Forecast (p. 15)

Sign Up for AWS

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all services in AWS, including Amazon Forecast. You are charged only for the services that you use.

If you already have an AWS account, skip to the next task. If you don't have an AWS account, use the following procedure to create one.

To sign up for AWS

1. Open https://aws.amazon.com/, and then choose **Create an AWS Account**.
2. Follow the on-screen instructions to complete the account creation. Note your 12-digit AWS account number. Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.
3. Create an AWS Identity and Access Management (IAM) admin user. See Creating Your First IAM User and Group in the *AWS Identity and Access Management User Guide* for instructions.

Set Up the AWS CLI

The AWS Command Line Interface (AWS CLI) is a unified developer tool for managing AWS services, including Amazon Forecast. We recommend that you install and use it.

1. To install the AWS CLI, follow the instructions in Installing the AWS Command Line Interface in the *AWS Command Line Interface User Guide*.
2. To configure the AWS CLI and set up a profile to call it, follow the instructions in Configuring the AWS CLI in the *AWS Command Line Interface User Guide*.
3. To confirm that the AWS CLI profile is configured correctly, run the following command in a command window:

```
aws configure --profile default
```

If your profile has been configured correctly, you should see output similar to the following:

```
AWS Access Key ID [****************52FQ]:
AWS Secret Access Key [****************xgyZ]:
Default region name [us-west-2]:
Default output format [json]:
```

4. To verify that the AWS CLI is configured for use with Amazon Forecast, run the following commands.

```
aws forecast help
```

```
aws forecastquery help
```

If the AWS CLI is configured correctly, you will see a list of the supported CLI commands for Amazon Forecast or Amazon Forecast Query.

Set Up Permissions for Amazon Forecast

Amazon Forecast uses Amazon Simple Storage Service (Amazon S3) to store the target time-series data that are used to train predictors that can generate forecasts. To access Amazon S3 on your behalf, Amazon Forecast needs your permission.

To grant Amazon Forecast permission to use Amazon S3 on your behalf, you must have an AWS Identity and Access Management (IAM) role and IAM policy in your account. The IAM policy specifies the required permissions, and must be attached to the IAM role.

To create the IAM role and policy and to attach the policy to the role, you can use the IAM console or the AWS Command Line Interface (AWS CLI).

Topics
- Create an IAM Role for Amazon Forecast (IAM Console) (p. 15)
- Create an IAM for Amazon Forecast (AWS CLI) (p. 16)

Create an IAM Role for Amazon Forecast (IAM Console)

You can use the AWS IAM console to do the following:

- Create an IAM role with Amazon Forecast as a trusted entity
- Create an IAM policy with permissions that allows Amazon Forecast to show, read, and write data in an Amazon S3 bucket
- Attach the IAM policy to the IAM role

To create an IAM role and policy that allows Amazon Forecast to access Amazon S3 (IAM console)

1. Sign in to the IAM console (https://console.aws.amazon.com/iam).
2. Choose **Policies** and do the following to create the required policy:

   a. On the **Create policy** page, in the policy editor, choose the **JSON** tab.
   b. Copy the following policy and replace the text in the editor by pasting the this policy over it. Be sure to replace *bucket-name* with the name of your S3 bucket, then choose **Review policy**.

   ```
   {
       "Version":"2012-10-17",
       "Statement":[
           {
               "Effect":"Allow",
               "Action":[
   ```

```
                "s3:Get*",
                "s3:List*",
                "s3:PutObject"
            ],
            "Resource":[
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
}
```

    c.   In **Review policy**, for **Name**, enter a name for the policy. For example, `AWSS3BucketAccess`. Optionally, provide a description for this policy, then choose **Create policy**.

3.   In the navigation pane, choose **Roles**. Then do the following to create the IAM role:

    a.   Choose **Create role**.

    b.   For **Select type of trusted entity**, choose **AWS service**.

    c.   For **Choose the service that will use this role**, if you don't see **Amazon Forecast** listed, choose **EC2**. Otherwise, choose **Amazon Forecast**.

    d.   Choose **Next: Permissions**.

    e.   For **Attach permissions policies**, choose the check box next to the policy that you just created. To display the policy in the list, type part of your policy name in the **Filter policies** query filter. Then, choose **Next: Tags**.

    f.   You don't need to add tags, so choose **Next: Review**.

    g.   In the **Review** section, for **Role name**, enter a name for the role (for example, `ForecastRole`). Update the description for the role in **Role description**, then choose **Create role**.

    h.   Choose the new role to open the role's details page.

    i.   In the **Summary**, copy the **Role ARN** value and save it. You need it to import a dataset into Amazon Forecast.

    j.   If you didn't choose **Amazon Forecast** as the service that will use this role, choose **Trust relationships**, and then choose **Edit trust relationship** to update the trust policy as follows.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "forecast.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Create an IAM for Amazon Forecast (AWS CLI)

You can use the AWS CLI to do the following:

- Create an IAM role with Amazon Forecast as a trusted entity
- Create an IAM policy with permissions that allows Amazon Forecast to show, read, and write data in an Amazon S3 bucket
- Attach the IAM policy to the IAM role

To create an IAM role and policy that allows Amazon Forecast to access Amazon S3 (AWS CLI)

1. Create an IAM role with Amazon Forecast as a trusted entity that can assume the role for you:

```
aws iam create-role \
 --role-name ForecastRole \
 --assume-role-policy-document '{
    "Version":"2012-10-17",
    "Statement":[
        {
           "Effect":"Allow",
           "Principal":{
              "Service":"forecast.amazonaws.com"
           },
           "Action":"sts:AssumeRole"
        }
    ]
}
```

   This command assumes that the default AWS configuration profile is targeted for an AWS Region supported by Amazon Forecast. If you have configured another profile (for example, `aws-forecast`) to target an AWS Region that is not supported by Amazon Forecast, you must explicitly specify that configuration by including the `profile` parameter in the command, for example, `--profile aws-forecast`. For more information about setting up an AWS CLI configuration profile, see the AWS CLI configure command.

   If the command successfully creates the role, it returns it as output, which should look similar to the following:

```
{
    "Role": {
        "RoleName": "ForecastRole",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": "sts:AssumeRole",
                    "Principal": {
                        "Service": "forecast.amazonaws.com"
                    },
                    "Effect": "Allow"
                }
            ]
        },
        "Arn": "arn:aws:iam::your-acct-ID:role/ForecastRole",
        "CreateDate": "2018-09-12T00:23:06Z",
        "RoleId": "AROAITEGTQ3NN3FYHXNJU",
        "Path": "/"
    }
}
```

   Record the role's ARN. You need it when you import a dataset to train an Amazon Forecast predictor.

2. Create an IAM policy with permissions to list, read, and write data in Amazon S3, and attach it to the IAM role that you created in Step 1:

```
aws iam put-role-policy \
  --role-name ForecastRole \
  --policy-name ForecastBucketAccessPolicy \
  --policy-document '{
   "Version":"2012-10-17",
```

```
    "Statement":[
        {
            "Effect":"Allow",
            "Action":[
                "s3:Get*",
                "s3:List*",
                "s3:PutObject"
            ],
            "Resource":[
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
}'
```

Getting Started

To get started using Amazon Forecast, you do the following.

- Create an Amazon Forecast dataset and import training data.
- Create an Amazon Forecast predictor. The algorithm that you choose, trains a predictor using the datasets. You specify both the algorithm and dataset when you create the predictor.
- Generate a forecast.

In this exercise, you use a modified version of a publicly available electricity usage dataset to train predictors. For more information, see ElectricityLoadDiagrams20112014 Data Set. The following are sample rows from the dataset:

```
2014-01-01 01:00:00,    2.53807106598985, client_0
2014-01-01 01:00:00, 23.648648648648624, client_1
2014-01-01 02:00:00,  9.648648648612345, client_0
```

For this exercise, you use the dataset to train a predictor, and then predict the hourly electricity usage by client.

You can use either the Amazon Forecast console or the AWS Command Line Interface (AWS CLI) for this exercise. Pay attention to the default regions of the Amazon Forecast console, the AWS CLI, and the Amazon Forecast SDKs, as Amazon Forecast resources are not shared across regions.

Important
Before you begin, make sure that you have an AWS account and have installed the AWS CLI. For more information, see Setting Up (p. 14). We also recommend that you review How Amazon Forecast Works (p. 2).

Topics

Prepare Input Data

Regardless of whether you use the Amazon Forecast console or the AWS Command Line Interface (AWS CLI) to set up a forecasting project, you need to set up your input data. To prepare your data, you do the following:

- Download training data to your computer and upload it to an Amazon Simple Storage Service (Amazon S3) bucket in your AWS account. To import your data to an Amazon Forecast dataset, you must store it in an Amazon S3 bucket.
- Create an AWS Identity and Access Management (IAM) role. You give Amazon Forecast permission to access your S3 bucket with the IAM role. For more information about IAM roles, see IAM Roles in the *IAM User Guide*.

To prepare training data

1. Download the zip file, electricityusagedata.zip.

For this exercise, you use the individual household electric power consumption dataset. (Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.) We aggregate the usage data hourly.

2. Unzip the content and save it locally as `electricityusagedata.csv`.

3. Upload the data file to an S3 bucket.

   For step-by-step instructions, see Uploading Files and Folders by Using Drag and Drop in the *Amazon Simple Storage Service Console User Guide.*

4. Create an IAM role.

   If you want to use the AWS CLI for the Getting Started exercise, you must create an IAM role. If you use the console, you can have it create the role for you. For step-by-step instructions, see Set Up Permissions for Amazon Forecast (p. 15).

Now, use the Amazon Forecast console or the AWS CLI to train a predictor, generate a forecast, and see the forecast.

- Getting Started (Console) (p. 20)
- Getting Started (AWS CLI) (p. 35)

Getting Started (Console)

In this exercise, you use the Amazon Forecast console to import time-series data of electricity usage, create an Amazon Forecast predictor based on the input dataset, and make predictions of future electricity usage based on the input time interval.

For this exercise, we use the individual household electric power consumption dataset. (Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.) We aggregate the usage data hourly.

Prerequisites

- An AWS account. If you don't already have an AWS account, create one as described in Sign Up for AWS (p. 14).
- Training data in your Amazon Simple Storage Service (Amazon S3) bucket. For more information, see Prepare Input Data (p. 19).
- An AWS Identity and Access Management (IAM) role that allows Amazon Forecast to read and write to your S3 buckets. For more information, see Create an IAM Role for Amazon Forecast (IAM Console) (p. 15).

Step 1: Import Training Data

To import time-series data into Amazon Forecast, create a dataset group, choose a domain for your dataset group, specify the details of your data, and point Amazon Forecast to the S3 location of your data. You use a time series of historical electricity usage (p. 19) as an example for the target time series data.

Note
This exercise assumes that you haven't created any dataset groups. If you previously created a dataset group, what you see will vary slightly from the following screenshots and instructions.

To import time-series data for forecasting

1. Sign in to the AWS Management Console and open the Amazon Forecast console at https://console.aws.amazon.com/forecast/.

2. On the Amazon Forecast home page, choose **Create dataset group**.

3. On the **Create dataset group** page, for **Dataset group details**, provide the following information:

   - **Dataset group name** – Enter a name for your dataset group.

   - **Forecasting domain** – From the drop-down menu, choose **Custom**. For more information about how to choose a forecasting domain, see *How Amazon Forecast Works* (p. 2) and dataset domains and types (p. 46).

   Your screen should look similar to the following:

   ## Create dataset group Info

   Dataset groups are containers for all your datasets.

   ### Dataset group details

   Dataset group name
   The name that you enter here can help you distinguish this dataset group from other dataset groups on the Dataset groups dashboard.

   > my_dsgroup

   The dataset group name must have 1 to 32 characters. Valid characters: a-z, A-Z, 0-9, and . : + = @ _ %

   Forecasting domain Info
   A forecasting domain defines a forecasting use case. You can choose a predefined domain, or you can create your own domain.

   > Custom ▼
   > Choose this domain if none of the other domains are applicable to your forecas…

   Cancel   **Next**

4. Choose **Next**.

5. On the **Create target time series dataset** page, for **Dataset details**, provide the following information:

   - **Dataset name** – Enter a name for your dataset.

   - **Frequency of your data** – Keep the default value of **1**, and choose **hour** from the drop-down menu. This setting must be consistent with the input time series data. The time interval in the sample electricity-usage data is an hour.

   - **Data schema** – Update the schema to match the columns of the time-series data in data types and order. For the electricity usage input data, the columns correspond to: a timestamp, the electricity usage at the specified time (target_value), and the ID of the customer charged for the electricity usage (string), in that order.

   Your screen should look similar to the following:

6. Choose **Next**.

7. On the **Import target time series data** page, for **Dataset import job details**, provide the following information:

    - **Dataset import job name** – Enter a name for your dataset.

    - **Timestamp format** – Leave the default (`yyyy-MM-dd HH:mm:ss`). The format must be consistent with the input time series data.

    - **IAM role** – Keep the default **Enter a custom IAM role ARN**.

      Alternatively, you can have Amazon Forecast create the required IAM role for you by choosing **Create a new role** from the drop-down menu and following the on-screen instructions.

    - **Custom IAM role ARN** – Enter the Amazon Resource Name (ARN) of the IAM role that you created in Create an IAM Role for Amazon Forecast (IAM Console) (p. 15).

    - **Data location** – Use the following format to enter the location of your .csv file on Amazon S3:

```
s3://<name of your S3 bucket>/<folder path>/<filename.csv>
```

Your screen should look similar to the following:

8. Choose **Start import**.
9. The dataset group's **Dashboard** page is displayed. Your screen should look similar to the following:

Under **Target time series data**, you will see the status of the import job. Wait for Amazon Forecast to finish importing your time-series data. The process can take several minutes or longer. When your dataset has been imported, the status transitions to **Active**. Additionally, the banner at the top of the dashboard, changes to display the following message:

Now that your target time series dataset has been imported, you can train a predictor.

Step 2: Train a Predictor

To create a predictor, which is a trained model, choose an algorithm and the number (length times frequency) of predictions to make. You can choose a particular algorithm, or you can choose **AutoML** to have Amazon Forecast process your data and choose an algorithm to best suit your dataset group. For information about algorithms, see *Choosing an Algorithm* (p. 55).

To train a predictor

1. After your target time series dataset has finished importing, your dataset group's **Dashboard** should look similar to the following:
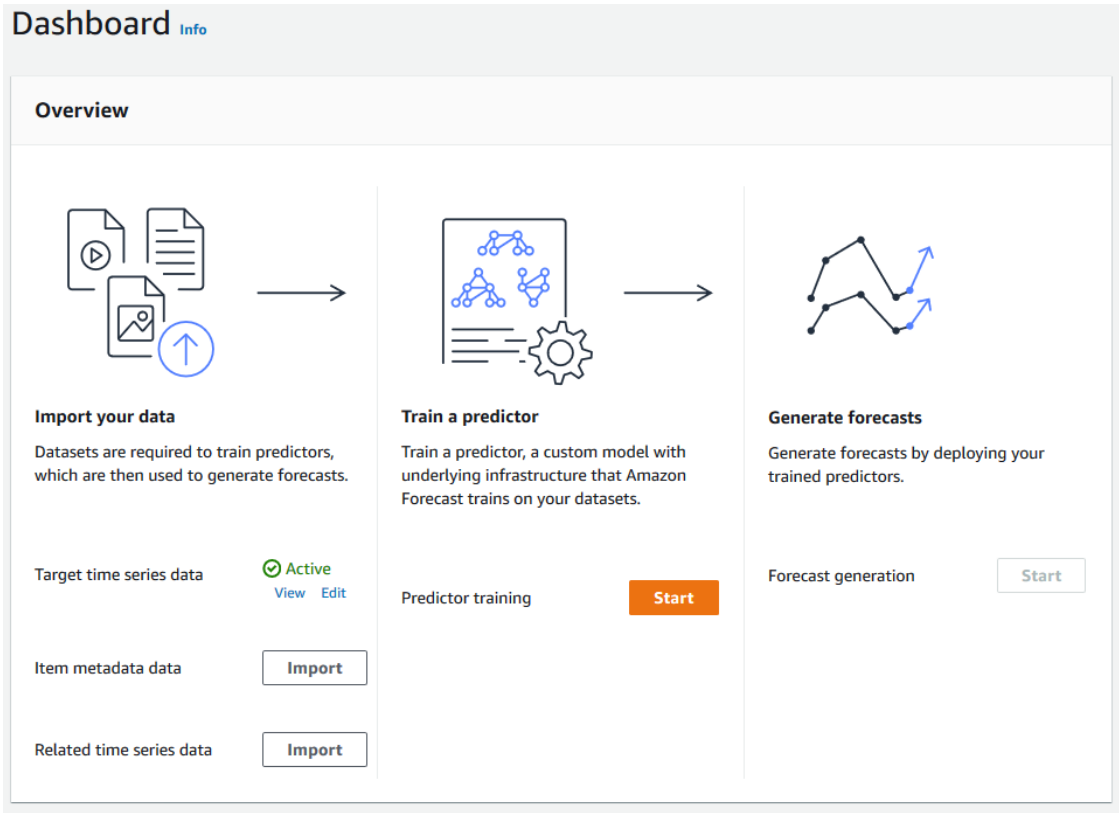
Under **Train a predictor**, choose **Start**. The **Train predictor** page is displayed.

> **Note**
> The `Status` of the **Target time series data** must be `Active`, which signifies that the import successfully finished, before you can train the predictor.

2. On the **Train predictor** page, for **Predictor details**, provide the following information:

    *   **Predictor name** – Enter a name for your predictor.

    *   **Forecast horizon** – Choose how far into the future to make predictions. This number multiplied by the data entry frequency (`hourly`) that you specified in `Step 1: Import the Training Data` determines how far into the future to make predictions. For this exercise, set the number to `36`, to provide predictions for 36 hours.

    *   **Forecast frequency** – Keep the default value of **1**. From the drop-down menu, choose **hour**. This setting must be consistent with the input time series data. The time interval in the sample electricity-usage data is an hour.

- **Algorithm selection** – Keep the default value **Manual**. From the drop-down menu, choose the **ETS** algorithm. For more information about recipes, see Choosing an Amazon Forecast Algorithm (p. 55).

The remaining settings are optional, so leave the default values. Your screen should look similar to the following:

Predictor details

Predictor name
The name that you enter here can help you distinguish this predictor from your other predictors.

```
my_predictor
```

The predictor name must have 1 to 32 characters. Valid characters: a-z, A-Z, 0-9, and . : + = @ _ %

Forecast horizon Info
The range tells Amazon Forecast how far into the future to forecast your data. The number you enter here will be multiplied by the data update interval of your target time-series dataset.

```
36
```

Forecast frequency
This is the frequency at which your forecasts are generated.

Your forecast frequency should be | 1 ▼ | hour ▼ |

Algorithm selection Info
An algorithm is used to train your predictor.

○ **Automatic (AutoML)**
Let Amazon Forecast choose the right algorithm for your dataset.

◉ **Manual**
Explore the algorithms and choose one.

Algorithm
The algorithm that you want Amazon Forecast to use to train your predictor.

```
ETS
arn:aws:forecast:::algorithm/ETS                          ▼
```

Forecast dimensions - optional
Item id is used in training by default. Select additional keys you would like to use to generate a forecast. These keys are fields in your dataset.

```
Select a forecast dimension                               ▼
```

Country for holidays - optional
The holiday calendar you want to include for model training

```
Choose a country                                          ▼
```

Number of backtest windows - optional Info
This is the number of times that the algorithm splits the input data for use in training and evaluation.

```
1
```

Backtest window offset - optional Info
This is the point in the dataset where you want to split the data for model training and evaluation.

```
36
```

Training subsample ratio - optional
This is the percentage of items in the data that you want Amazon Forecast to use for training. This is a value greater than 0 and less than or equal to 1.

```
1
```

▶ **Advanced configurations**
Set advanced configurations for your predictor and forecasts.

3. Choose **Train predictor**. Your dataset group's **Dashboard** page is displayed. Your screen should look similar to the following:

Under **Predictor training**, you will see the training status. Wait for Amazon Forecast to finish training the predictor. The process can take several minutes or longer. When your predictor has been trained, the status transitions to **Active**. Additionally, the banner at the top of the dashboard changes to display the following message:

Now that your predictor has been trained, you can create a forecast.

Step 3: Create a Forecast

To make predictions (inferences), you use a predictor to create a forecast. A forecast is a group of predictions, one for every item in the target dataset. To retrieve the prediction for a single item, you query the forecast. To retrieve the complete forecast, you create an export job.

To get and view your forecast

1. After your predictor has finished training, your dataset group's **Dashboard** should look similar to the following:
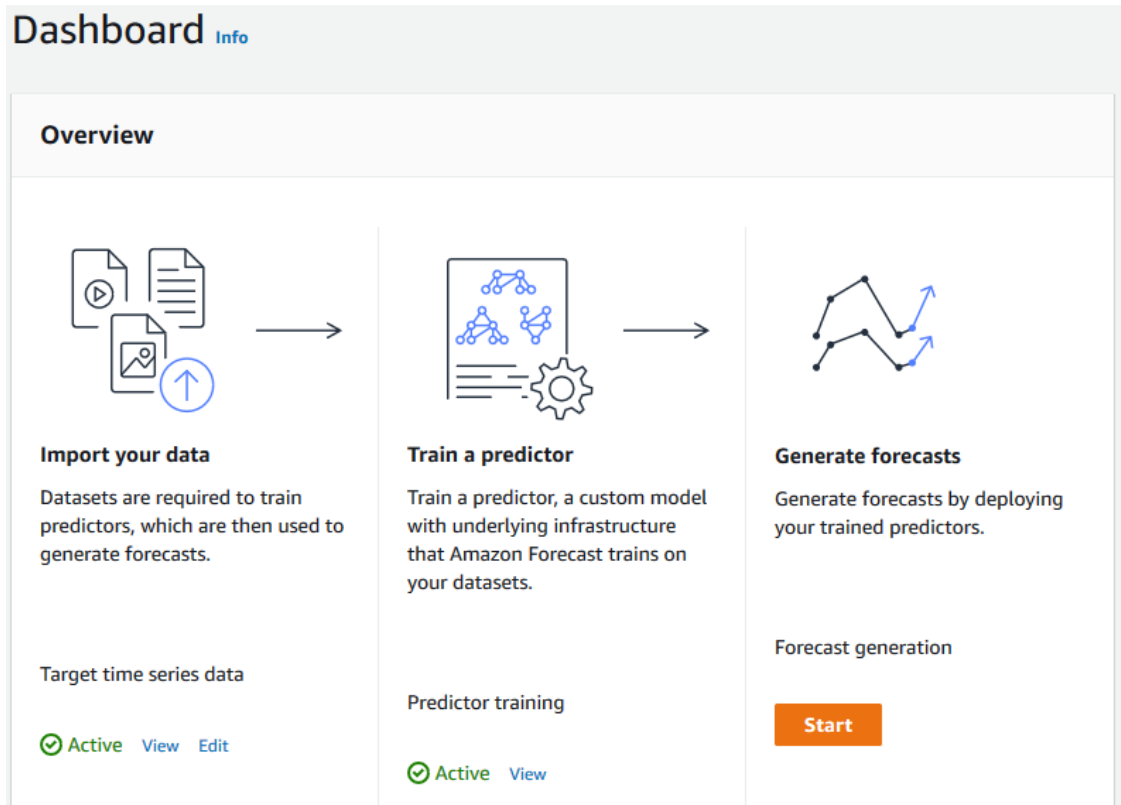
Under **Forecast generation**, choose **Start**. The **Create a forecast** page is displayed.

> **Note**
> The `Status` of **Predictor training** must be `Active` before you can generate a forecast.

2. On the **Create a forecast** page, for **Forecast details**, provide the following information:

   - **Forecast name** – Enter a name for your forecast.

   - **Predictor** – From the drop-down menu, choose the predictor that you created in `Step 2: Train a Predictor`.

   The remaining setting is optional, so leave the default value. Your screen should look similar to the following:

3. Choose **Create a forecast**. The dataset group's **Dashboard** page is displayed. Your screen should look similar to the following:

Under **Forecast generation**, you should see the status of forecast generation. Wait for Amazon Forecast to finish creating the forecast. The process can take several minutes or longer. When your forecast has been created, the progress transitions to **Active**. Additionally, the banner at the top of the dashboard changes to display the following message:

⊘ **Your forecast my_forecast is created**
You can view the forecasts for your items on the Forecast lookup page.
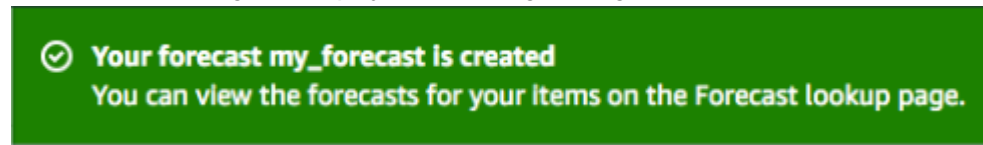
Now that your forecast has been created, you can query or export the forecast.

Step 4: Retrieve a Forecast

After the forecast has been created, you can query for a single item or export the complete forecast.

To query for a single item

1. If the dashboard is not displayed, in the navigation pane, under your dataset group, choose **Dashboard**.
2. In the Dashboard, under **Generate forecasts**, choose **Lookup forecast**. The **Forecast lookup** page is displayed.
3. On the **Forecast lookup** page, for **Forecast details**, provide the following information.

    *   **Forecast** – From the drop-down menu, choose the forecast that you created in `Step 3: Create a Forecast`.
    *   **Start date** – Enter **2015/01/01**. Keep the default time of `00:00:00`.
    *   **End date** – Enter **2015/01/02**. Change the time to `12:00:00`.

        The date range of 36 hours corresponds to the **Forecast horizon** that you specified in `Step 2: Train a Predictor`.
    *   **Choose which keys/filters** – Choose **Add forecast key**.
    *   **Forecast key** – From the drop-down menu, choose `item_id`.
    *   **Value** – Enter a value from the `item_id` column of the input time series of the electricity usage data. An `item_id` (for example, `client_21`) identifies a particular client who is included in the dataset.

    Your screen should look similar to the following:

4. Choose **Get Forecast**. When the forecast is displayed, review the forecast for electricity usage demand by `client_21`.

   The forecast should look similar to the following:
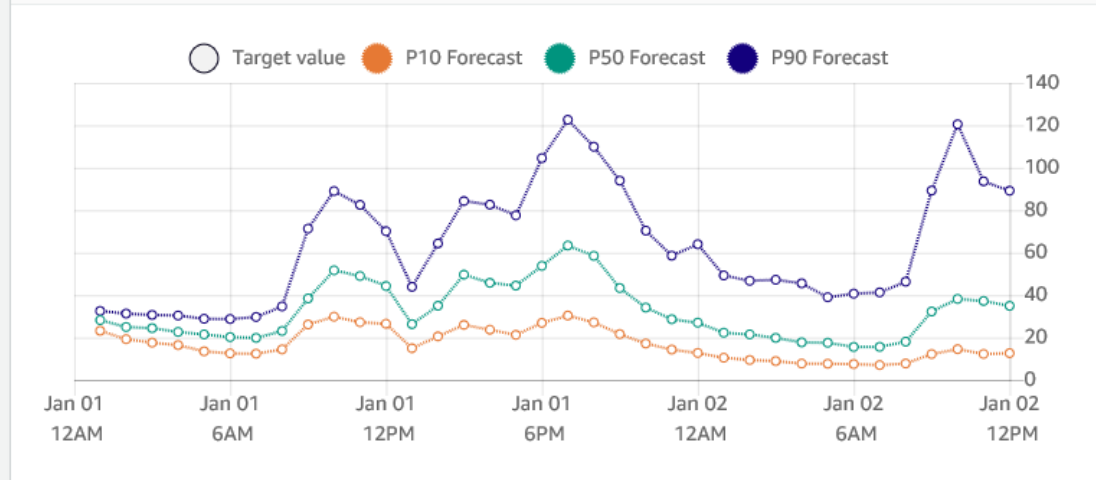
To export the complete forecast

1. In the navigation pane, under your dataset group, choose **Forecasts**.

2. Choose the radio button next to the forecast that you created in `Step 3: Create a Forecast.`

3. Choose **Create forecast export**. The **Create forecast export** page is displayed.

4. On the **Create forecast export** page, for **Export details**, provide the following information.

   - **Export name** – Enter a name for your forecast export job.

   - **Generated forecast** – From the drop-down menu, choose the forecast that you created in `Step 3: Create a Forecast.`

   - **IAM role** – Keep the default **Enter a custom IAM role ARN**.

     Alternatively, you can have Amazon Forecast create the required IAM role for you by choosing **Create a new role** from the drop-down menu and following the on-screen instructions.

   - **Custom IAM role ARN** – Enter the Amazon Resource Name (ARN) of the IAM role that you created in Create an IAM Role for Amazon Forecast (IAM Console) (p. 15).

   - **S3 forecast export location** – Use the following format to enter the location of your Amazon Simple Storage Service (Amazon S3) bucket or folder in the bucket:

     `s3://<name of your S3 bucket>/<folder path>/`

   Your screen should look similar to the following:

5. Choose **Create forecast export**. The **my_forecast** page is displayed.

   Your screen should look similar to the following:

   You should see the status progress. Wait for Amazon Forecast to finish exporting the forecast. The process can take several minutes or longer. When your forecast has been exported, the status transitions to **Active** and you can find the forecast files in your S3 bucket.

Getting Started (AWS CLI)

In this exercise, you use the AWS Command Line Interface (CLI) to explore Amazon Forecast. You create an Amazon Forecast dataset, train a predictor, and use the resulting predictor to generate a forecast. Before you begin, make sure that you have an AWS account and that you've set up the AWS CLI. For more information, see Setting Up (p. 14).

> **Note**
> The CLI commands in this exercise were tested on Linux. For information about using the CLI commands on Windows, see Specifying Parameter Values for the AWS Command Line Interface in the *AWS Command Line Interface User Guide*.

Step 1: Import Training Data

Begin by creating a dataset and importing the electricity usage data into it.

To create an Amazon Forecast dataset

1. Decide which domain and dataset type is appropriate.

   The training data that you will import into the dataset influences your choice of dataset domain and type. So, let's review a few sample rows of the electricity usage data.

   ```
   2014-01-01 01:00:00,    2.53807106598985, client_0
   2014-01-01 01:00:00, 23.648648648648624, client_1
   2014-01-01 02:00:00,  9.648648648612345, client_0
   ```

   The data format is CSV (comma-separated values), and it's collected hourly (as shown by the timestamps). It includes these columns:

   - Column 1 – Timestamps that show when electricity usage was recorded.
   - Column 2 – Hourly electricity usage values (note how the timestamp values increase by hour).
   - Column 3 – Client ID values that identify the customers using the electricity.

   For this data, choose the following predefined dataset domain and dataset type:

   - Custom domain – None of the dataset domains, such as METRICS, RETAIL, or WEB_TRAFFIC, applies to this data, so choose the Custom domain.
   - Target time series type – The data is a time series because it tracks electricity usage over time. It also includes the *target* that we want to forecast (Column 2, electricity usage). Therefore, choose the target time series dataset type.

     To understand why you choose this type, see Predefined Dataset Domains and Dataset Types (p. 46).

2. Decide on a dataset schema.

   The target time series type for the CUSTOM Domain (p. 49) requires these fields; `timestamp`, `target_value`, and `item_id`. The `target_value` field is the target. Amazon Forecast generates the forecast for this field.

   To map the required fields to columns in your data, you create a schema. Each *attribute* in the schema maps to a field in the data.

   > **Important**
   > The order of attributes in the schema must match the order of fields in the training data.

```
{
  "Attributes":[
    {
      "AttributeName": "timestamp",
      "AttributeType": "timestamp"
    },
    {
      "AttributeName": "target_value",
      "AttributeType": "float"
    },
    {
      "AttributeName": "item_id",
      "AttributeType": "string"
    }
  ]
}
```

You now have the information necessary to create a dataset and import data into it.

3. Create the dataset.

```
aws forecast create-dataset \
--dataset-name electricity_demand_ds \
--domain CUSTOM \
--dataset-type TARGET_TIME_SERIES \
--data-frequency H \
--schema '{
  "Attributes": [
    {
      "AttributeName": "timestamp",
      "AttributeType": "timestamp"
    },
    {
      "AttributeName": "target_value",
      "AttributeType": "float"
    },
    {
      "AttributeName": "item_id",
      "AttributeType": "string"
    }
  ]
}'
```

In the request, the `data-frequency` value `H` represents a data collection frequency of hourly. The following is an example response.

```
{
    "DatasetArn": "arn:aws:forecast:us-west-2:acct-id:dataset/electricity_demand_ds"
}
```

For more information about this operation, see CreateDataset (p. 113).

4. (Optional) Get the description of the dataset.

```
aws forecast describe-dataset \
--dataset-arn arn:aws:forecast:us-west-2:acct-id:dataset/electricity_demand_ds
```

The following is an example response.

```
{
```

```
    "DatasetName": "electricity_demand_ds",
    "DatasetArn": "arn:aws:forecast:us-west-2:acct-id:dataset/electricity_demand_ds",
    "CreationTime": 1564533087.907,
    "LastModificationTime": 1564533087.907,
    "Domain": "CUSTOM",
    "DatasetType": "TARGET_TIME_SERIES",
    "DataFrequency": "H",
    "Schema": { ... },
    "EncryptionConfig": {},
    "Status": "ACTIVE"
}
```

Note
The order of the key-value pairs in the response is arbitrary.

5. Create a dataset group and add the dataset to it. The value of the `domain` parameter must match the `domain` of the dataset.

```
aws forecast create-dataset-group \
--dataset-group-name electricity_ds_group \
--dataset-arns arn:aws:forecast:us-west-2:acct-id:ds/electricity_demand_ds \
--domain CUSTOM
```

The following is an example response.

```
{
    "DatasetGroupArn": "arn:aws:forecast:us-west-2:acct-id:dataset-group/
electricity_ds_group"
}
```

For more information about this operation, see CreateDatasetGroup (p. 117).

6. (Optional) Get the description of the dataset group.

```
aws forecast describe-dataset-group \
--dataset-group-arn arn:aws:forecast:us-west-2:acct-id:dataset-group/
electricity_ds_group
```

The following is an example response.

```
{
    "DatasetGroupName": "electricity_ds_group",
    "DatasetGroupArn": "arn:aws:forecast:us-west-2:acct-id:dataset-group/
electricity_ds_group",
    "DatasetArns": [
        "arn:aws:forecast:us-west-2:acct-id:dataset-group/electricity_ds_group"
    ],
    "Domain": "CUSTOM",
    "CreationTime": 1564533719.852,
    "LastModificationTime": 1564533719.852,
    "Status": "ACTIVE"
}
```

7. Import the electricity usage training data from your Amazon S3 bucket to the dataset. The IAM role that you provide must have permission to read data from your S3 bucket. For information on how to create an IAM role, see Create an IAM for Amazon Forecast (AWS CLI) (p. 16).

```
aws forecast create-dataset-import-job \
--dataset-arn arn:aws:forecast:us-west-2:acct-id:dataset/electricity_demand_ds \
--dataset-import-job-name electricity_ds_import_job \
```

```
--data-source '{
    "S3Config": {
      "Path": "s3://bucket/electricityusagedata.csv",
      "RoleArn": "arn:aws:iam::acct-id:role/Role"
    }
 }'
```

The following is the shorthand syntax for the `data-source` parameter.

```
--data-source S3Config="{Path='s3://bucket/
electricityusagedata.csv',RoleArn='arn:aws:iam::acct-id:role/Role'}"
```

The following is an example response.

```
{
    "DatasetImportJobArn": "arn:aws:forecast:us-west-2:acct-id:dataset-import-job/
electricity_demand_ds/electricity_ds_import_job"
}
```

For more information about this operation, see CreateDatasetImportJob (p. 120).

8. Check the import status.

```
aws forecast describe-dataset-import-job \
--dataset-import-job-arn arn:aws:forecast:us-west-2:acct-id:dataset-import-job/
electricity_demand_ds/electricity_ds_import_job
```

The following is an example response.

```
{
    "DatasetImportJobName": "electricity_ds_import_job",
    "DatasetImportJobArn": "arn:aws:forecast:us-west-2:acct-id:dataset-import-job/
electricity_demand_ds/electricity_ds_import_job",
    "DatasetArn": "arn:aws:forecast:us-west-2:acct-id:dataset/electricity_demand_ds",
    "DataSource": {
        "S3Config": {
            "Path": "s3://bucket/electricityusagedata.csv",
            "RoleArn": "arn:aws:iam::acct-id:role/ForecastRole"
        }
    },
    "DataSize": 0.14639010466635227,
    "TimeStampFormat": "yyyy-MM-dd HH:mm:ss",
    "CreationTime":  1564537011.114,
    "LastModificationTime": 1564537028.223,
    "Status": "CREATE_IN_PROGRESS"
}
```

When all of the data has been imported, the status changes to ACTIVE and the response includes statistics for the data, as shown in the following example.

```
{
    "DatasetArn": "arn:aws:forecast:us-west-2:acct-id:dataset/electricity_demand_ds",
    "Status": "ACTIVE",
    "FieldStatistics": {
        "date": {
            "Min": "2014-01-01T01:00:00Z",
            "Max": "2015-01-01T00:00:00Z",
            "Count": 3241200,
            "CountDistinct": 8760,
```

```
            "CountNull": 0
        },
        "target": {
            "Min": "0.0",
            "Max": "168200.0",
            "Avg": 606.5167610461679,
            "Stddev": 3518.405223972031,
            "Count": 3241200,
            "CountDistinct": 1196961,
            "CountNull": 0,
            "CountNan": 0
        },
        "item": {
            "Count": 3241200,
            "CountDistinct": 370,
            "CountNull": 0
        }
    },
    ...
}
```

Important
You must wait until the status is ACTIVE before creating a predictor with the dataset group.

For more information about this operation, see DescribeDatasetImportJob (p. 154).

Step 2: Train a Predictor

To create a predictor, you use the CreatePredictor (p. 129) operation and provide the following information.

- An algorithm – Amazon Forecast uses the algorithm to train the predictor using the data in the dataset group. For this exercise, you use an algorithm called `forecast_DEEP_AR_PLUS`, which is provided by Amazon Forecast. For a list of algorithms that Amazon Forecast provides, see Choosing an Amazon Forecast Algorithm (p. 55).
  Note
  If you aren't sure which algorithm to use, you can set the `PerformAutoML` flag in the `CreatePredictor` operation to tell Amazon Forecast to run AutoML. AutoML determines which algorithm to use for predictor training.
- A dataset group – You created the dataset group in the preceding step.

After the predictor is created, you review the accuracy metrics generated by Amazon Forecast. The metrics help you decide whether to use the predictor for generating a forecast. For more information about predictors, see Predictors (p. 12).

To create a predictor and review the accuracy metrics

1. Create the predictor.

```
aws forecast create-predictor \
--predictor-name electricitypredictor \
--algorithm-arn arn:aws:forecast:::algorithm/Deep_AR_Plus \
--input-data-config DatasetGroupArn="arn:aws:forecast:us-west-2:acct-id:dsgroup/
electricity_ds_group" \
--forecast-horizon 20 \
--featurization-config '{
    "ForecastFrequency": "H"
  }'
```

The following is an example response.

```
{
    "PredictorArn": "arn:aws:forecast:us-west-2:acct-id:predictor/electricitypredictor"
}
```

2. Get the predictor's status.

```
aws forecast describe-predictor \
--predictor-arn arn:aws:forecast:us-west-2:acct-id:predictor/electricitypredictor
```

The following is an example response.

```
{
    "PredictorName": "electricitypredictor",
    "PredictorArn": "arn:aws:forecast:us-west-2:acct-id:predictor/
electricitypredictor",
    "AlgorithmArn": "arn:aws:forecast:::algorithm/Deep_AR_Plus",
    "DatasetImportJobArns": [
        "arn:aws:forecast:us-west-2:acct-id:dataset-import-job/electricity_demand_ds/
electricity_ds_import_job"
    ],
    "InputDataConfig": {
        "DatasetGroupArn": "arn:aws:forecast:us-west-2:acct-id:dataset-group/
electricity_ds_group"
    },
    "ForecastHorizon": 20,
    "FeaturizationConfig": {
        "ForecastFrequency": "H",
        "Featurizations": [
            {
                "AttributeName": "target_value",
                "FeaturizationPipeline": [
                    {
                        "FeaturizationMethodName": "filling",
                        "FeaturizationMethodParameters": {
                            "frontfill": "none",
                            "aggregation": "sum",
                            "backfill": "zero",
                            "middlefill": "zero"
                        }
                    }
                ]
            }
        ]
    },
    "CreationTime": 1564611261.617,
    "LastModificationTime": 1564611279.896,
    "PerformAutoML": false,
    "PerformHPO": false,
    "EvaluationParameters": {
        "BackTestWindowOffset": 20,
        "NumberOfBacktestWindows": 1
    },
    "Status": "CREATE_IN_PROGRESS"
}
```

Important
Model training takes time. Don't proceed until training has completed and the status of the predictor is ACTIVE.

3. Get the accuracy metrics for the predictor.

```
aws forecast get-accuracy-metrics \
--predictor-arn arn:aws:forecast:us-west-2:acct-id:predictor/electricitypredictor
```

The following is an example response.

```
{
    "PredictorEvaluationResults": [
        {
            "TestWindows": [
                {
                    "EvaluationType": "SUMMARY",
                    "Metrics": {
                        "RMSE": 448.19602551622864,
                        "WeightedQuantileLosses": [
                            {
                                "Quantile": 0.9,
                                "LossValue": 0.11574311406253326
                            },
                            {
                                "Quantile": 0.5,
                                "LossValue": 0.1706269067283527
                            },
                            {
                                "Quantile": 0.1,
                                "LossValue": 0.11724164222477837
                            }
                        ]
                    }
                },
                {
                    "EvaluationType": "COMPUTED",
                    "Metrics": {
                        "RMSE": 448.19602551622864,
                        "WeightedQuantileLosses": [
                            {
                                "Quantile": 0.9,
                                "LossValue": 0.11574311406253326
                            },
                            {
                                "Quantile": 0.5,
                                "LossValue": 0.1706269067283527
                            },
                            {
                                "Quantile": 0.1,
                                "LossValue": 0.11724164222477837
                            }
                        ]
                    },
                    "TestWindowEnd":    1420070400.0,
                    "TestWindowStart": 1420002000.0
                }
            ]
        }
    ]
}
```

The metrics show the error loss for each quantile. For example, there was an 11.7% error for the first quantile. The metrics also show the root-mean-square error (`RMSE`).

The summary metrics show the average of the computed metrics over all test windows. Because there was only one test window, the summary and computed metrics are equal.

For more information about this operation, see GetAccuracyMetrics (p. 170).

Step 3: Create a Forecast

Amazon Forecast creates a forecast for the `target_value` field (as determined by the dataset domain and type) for each unique `item_id` in the dataset. In this exercise, the `target_value` field provides electricity usage and the `item_id` provides client IDs. You get a forecast for the hourly electricity usage by customer.

After the forecast has been created, you can query for a single item or export the complete forecast.

To create, retrieve, and export a forecast

1. Create the forecast.

    ```
    aws forecast create-forecast \
    --forecast-name electricityforecast \
    --predictor-arn arn:aws:forecast:us-west-2:acct-id:predictor/electricitypredictor
    ```

    The operation uses the predictor to create a forecast. In the response, you get the Amazon Resource Name (ARN) of the forecast. You use this ARN to retrieve and export the forecast. The following is an example response.

    ```
    {
        "ForecastArn": "arn:aws:forecast:us-west-2:acct-id:forecast/electricityforecast"
    }
    ```

    For more information about this operation, see CreateForecast (p. 123).

2. Retrieve the first two hours of the forecast for `client_1`.

    > **Note**
    > The service name, `forecastquery`, is different then the service name used elsewhere.

    ```
    aws forecastquery query-forecast \
    --forecast-arn arn:aws:forecast:us-west-2:acct-id:forecast/electricityforecast \
    --start-date 2015-01-01T00:00:00 \
    --end-date   2015-01-01T02:00:00 \
    --filters '{"item_id":"client_1"}'
    ```

    The operation includes the following parameters.

    *   `start-date` and `end-date` – Specifies an optional date range to retrieve the forecast for. If you don't specify these parameters, the operation returns the entire forecast for `client_1`.

    *   `filters` – Specifies the `item_id` filter to retrieve the electricity forecast for `client_1`.

        The following is the shorthand syntax for the `filters` parameter.

        ```
        --filters item_id="client_1"
        ```

    The following is an example response.

```
{
    "Forecast": {
        "Predictions": {
            "mean": [
                {
                    "Timestamp": "2015-01-01T01:00:00",
                    "Value": 20.952411651611328
                },
                {
                    "Timestamp": "2015-01-01T02:00:00",
                    "Value": 19.11078453063965
                }
            ],
            "p90": [
                {
                    "Timestamp": "2015-01-01T01:00:00",
                    "Value": 24.524038314819336
                },
                {
                    "Timestamp": "2015-01-01T02:00:00",
                    "Value": 22.319091796875
                }
            ],
            "p50": [
                {
                    "Timestamp": "2015-01-01T01:00:00",
                    "Value": 20.7841739654541
                },
                {
                    "Timestamp": "2015-01-01T02:00:00",
                    "Value": 19.237524032592773
                }
            ],
            "p10": [
                {
                    "Timestamp": "2015-01-01T01:00:00",
                    "Value": 18.507278442382812
                },
                {
                    "Timestamp": "2015-01-01T02:00:00",
                    "Value": 16.15062141418457
                }
            ]
        }
    }
}
```

Because this is an hourly forecast, the response shows hourly forecast values. In the response, note the following:

- `mean` – For the specific date and time, the mean is the predicted mean electricity usage value for the customer.

- `p90`, `p50`, and `p10` – Specify the confidence level that the actual value will be below the listed value at the specified date and time. For example, at 2015-01-01T01:00:00, Amazon Forecast is 90% confident that the electric usage will be below 24.5. Amazon Forecast is 50% confident that usage will be below 20.8, and 10% confident that usage will be below 18.5.

For more information about this operation, see QueryForecast (p. 190).

3. Export the complete forecast to your Amazon S3 bucket. The IAM role that you provide must have permission to write data to your S3 bucket. For information on how to create an IAM role, see Create an IAM for Amazon Forecast (AWS CLI) (p. 16).

Create a forecast export job.

```
aws forecast create-forecast-export-job \
--forecast-export-job-name electricityforecast_exportjob \
--forecast-arn arn:aws:forecast:us-west-2:acct-id:forecast/electricityforecast \
--destination S3Config="{Path='s3://bucket',RoleArn='arn:aws:iam::acct-id:role/Role'}"
```

The following is an example response.

```
{
    "ForecastExportJobArn": "arn:aws:forecast::us-west-2:acct-id:forecast-
export/64bbc087"
}
```

For more information about this operation, see CreateForecastExportJob (p. 126).

4. Get the status of the export job.

```
aws forecast describe-forecast-export-job \
--forecast-export-job-arn arn:aws:forecast:us-west-2:acct-id:forecast/
electricityforecast
```

The following is an example response.

```
{
    "ForecastExportJobArn": "arn:aws:forecast::us-west-2:acct-id:forecast-
export/64bbc087",
    "ForecastExportJobName": "electricityforecast_exportjob",
    "Status": "CREATE_IN_PROGRESS"
}
```

When the status is ACTIVE, you can find the forecast files in the specified S3 bucket.

Getting Started (Python Notebook)

To explore the Amazon Forecast APIs, you can use a Jupyter (Python) notebook. For information, see the Amazon Forecast samples on GitHub.

Clean Up Resources

To avoid incurring unnecessary charges, delete the resources you created after you're done with the getting started exercise. To delete the resources, use either the Amazon Forecast console or the `Delete` APIs from the SDKs or the AWS Command Line Interface (AWS CLI). For example, use the DeleteDataset (p. 135) API to delete a dataset.

To delete a resource, its status must be `ACTIVE`, `CREATE_FAILED`, or `UPDATE_FAILED`. Check the status using the `Describe` APIs, for example, DescribeDataset (p. 147).

Some resources must be deleted before others, as shown in the following table. This process can take some time.

To delete the training data you uploaded, `electricityusagedata.csv`, see How Do I Delete Objects from an S3 Bucket?.

Resource to Delete	Delete This First	Notes
`ForecastExportJob`		
`Forecast`		You can't delete a forecast while it is being exported. After a forecast is deleted, you can no longer query the forecast.
`Predictor`	All associated forecasts.	
`DatasetImportJob`		Can not be deleted.
`Dataset`		All `DatasetImportJobs` that target the dataset are also deleted. You can't delete a `Dataset` that is used by a predictor.
`DatasetSchema`	All datasets that reference the schema.	
`DatasetGroup`	All associated predictors All associated forecasts. All datasets in the dataset group.	You can't delete a `DatasetGroup` that contains a `Dataset` used by a predictor.

Predefined Dataset Domains and Dataset Types

To train a predictor, you create one or more datasets, add them to a dataset group, and provide the dataset group for training.

For each dataset that you create, you associate a dataset domain and a dataset type. A *dataset domain* defines a forecasting use case.

Amazon Forecast supports the following dataset domains:

- RETAIL Domain (p. 47) – For retail demand forecasting
- INVENTORY_PLANNING Domain (p. 50) – For supply chain and inventory planning
- EC2 CAPACITY Domain (p. 51) – For forecasting Amazon Elastic Compute Cloud (Amazon EC2) capacity
- WORK_FORCE Domain (p. 51) – For work force planning
- WEB_TRAFFIC Domain (p. 53) – For estimating future web traffic
- METRICS Domain (p. 53) – For forecasting metrics, such as revenue and cash flow
- CUSTOM Domain (p. 49) – For all other types of time-series forecasting

Each domain can have one to three *dataset types*. The dataset types that you create for a domain are based on the type of data that you have and what you want to include in training.

Each domain requires a target time series dataset, and optionally supports the related time series and item metadata dataset types.

The dataset types are:

- Target time series – The only required dataset type. This type defines the *target* field that you want to generate forecasts for. For example, if you want to forecast the sales for a set of products, then you must create a dataset of historical time-series data for each of the products that you want to forecast. Similarly, you can create a target time series dataset for metrics— such as revenue, cash flow, and sales —that you might want to forecast.
- Related time series – Time-series data that is related to the target time series data. For example, price is related to product sales data, so you might provide it as a related time series.
- Item metadata – Metadata that is applicable to the target time-series data. For example, if you are forecasting sales for a particular product, attributes of the product—such as brand, color, and genre —will be part of item metadata. When predicting EC2 capacity for EC2 instances, metadata might include the CPU and memory of the instance types.

For each dataset type, your input data must contain certain required fields. You can also include optional fields that Amazon Forecast suggests that you include.

The following examples show how to choose a dataset domain and corresponding dataset types.

Example Example 1: Dataset Types in the RETAIL Domain

If you are a retailer interested in forecasting demand for items, you might create the following datasets in the RETAIL domain:

- Target time series is the required dataset of historical time-series demand (sales) data for each item (each product a retailer sells). In the RETAIL domain, this dataset type requires that the dataset includes the `item_id`, `timestamp`, and the `demand` fields. The `demand` field is the forecast target, and is typically the number of items sold by the retailer in a particular week or day.
- Optionally, a dataset of the related time series type. In the RETAIL domain, this type can include optional, but suggested, time-series information such as `price`, `inventory_onhand`, and `webpage_hits`.
- Optionally, a dataset of the item metadata type. In the RETAIL domain, Amazon Forecast suggests providing metadata information related to the items that you provided in target time series, such as `brand`, `color`, `category`, and `genre`.

Example Example 2: Dataset Types in the METRICS Domain

If you want to forecast key metrics for your organization—such as revenue, sales and cash flow—you can provide Amazon Forecast with the following datasets:

- The target time series dataset that provides historical time-series data for the metric that you want to forecast. If your interest is to forecast the revenue of all of the business units in your organization, you can create a `target time series` dataset with the `metric`, `business unit`, and `metric_value` fields.
- If you have any metadata for each metric that isn't required, such as `category` or `location`, you might provide datasets of the related time series and item metadata type.

At a minimum, you must provide a target time series dataset for Forecast to generate forecasts for your target metrics.

Example Example 3: Dataset Types in the CUSTOM Domain

The training data for your forecasting application might not fit into any of the Amazon Forecast domains. If that's the case, choose the CUSTOM domain. You must provide the target time series dataset, but you can add your own custom fields.

The Getting Started (p. 19) exercise forecasts electricity usage for a client. The electricity usage training data doesn't fit into any of the dataset domains, so we used the CUSTOM domain. In the exercise, we use only one dataset type, the target time series type. We map the data fields to the minimum fields required by the dataset type.

RETAIL Domain

The RETAIL domain supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types (p. 3).

Topics

Target Time Series Dataset Type

The target time series is the historical time series data for each item or product sold by the retail organization. The following fields are required:

- `item_id` (string) – A unique identifier for the item or product that you want to predict the demand for.

- `timestamp` (timestamp)

- `demand` (float) – The number of sales for that item at the timestamp. This is also the *target* field for which Amazon Forecast generates a forecast.

The following dimension is optional and can be used to change forecasting granularity:

- `location` (string) – The location of the store that the item got sold at. This should only be used if you have multiple stores/locations.

Ideally, only these required fields and optional dimensions should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

You can provide Amazon Forecast with related time series datasets, such as the price or the number of web hits the item received on a particular date. The more information that you provide, the more accurate the forecast. The following fields are required:

- `item_id` (string)

- `timestamp` (timestamp)

The following fields are optional and might be useful in improving forecast results:

- `price` (float) – The price of the item at the time of the timestamp.

- `promotion_applied` (integer; 1=true, 0=false) – A flag that specifies whether there was a marketing promotion for that item at the timestamp.

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

This dataset provides Amazon Forecast with information about metadata (attributes) of the items whose demand is being forecast. The following fields are required:

- `item_id` (string)

The following fields are optional and might be useful in improving forecast results:

- `category` (string)

- `brand` (string)

- `color` (string)

- `genre` (string)

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

CUSTOM Domain

The CUSTOM domain supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types (p. 3).

Topics

- Target Time Series Dataset Type (p. 49)
- Related Time Series Dataset Type (p. 49)
- Item Metadata Dataset Type (p. 49)

Target Time Series Dataset Type

The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)
- `target_value` (floating-point integer) – This is the `target` field for which Amazon Forecast generates a forecast.

Ideally, only these required fields should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)

In addition to the required fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

The following field is required:

- `item_id` (string)

The following field is optional and might be useful in improving forecast results:

- `category` (string)

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

INVENTORY_PLANNING Domain

Use the INVENTORY_PLANNING domain for forecasting demand for raw materials and determining how much inventory of a particular item to stock. It supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types (p. 3).

Topics

Target Time Series Dataset Type

The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)
- `demand` (float) – This is the `target` field for which Amazon Forecast generates a forecast.

The following dimension is optional and can be used to change forecasting granularity:

- `location` (string) – The location of the distribution center where the item is stocked. This should only be used if you have multiple stores/locations.

Ideally, only these required fields and optional dimensions should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)

The following fields are optional and might be useful in improving forecast results:

- `price` (float) – The price of the item

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

The following fields are required:

- `item_id` (string)

The following fields are optional and might be useful in improving forecast results:

- `category` (string) – The category of the item.
- `brand` (string) – The brand of the item.
- `lead_time` (string) – The lead time, in days, to manufacture the item.
- `order_cycle` (string) – The order cycle starts when work begins and ends when the item is ready for delivery.
- `safety_stock` (string) – The minimum amount of stock to keep on hand for that item.

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

EC2 CAPACITY Domain

Use the EC2 CAPACITY domain for forecasting Amazon EC2 capacity. It supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types (p. 3).

Target Time Series Dataset Type

The following fields are required:

- `instance_type` (string) – The type of instance (for example, c5.xlarge).
- `timestamp` (timestamp)
- `number_of_instances` (integer) – The number of instances of that particular instance type that was consumed at the timestamp. This is the `target` field for which Amazon Forecast generates a forecast.

The following dimension is optional and can be used to change forecasting granularity:

- `location` (string) – You can provide an AWS Region, such as us-west-2 or us-east-1. This should only be used if you're modeling multiple Regions.

Ideally, only these required and suggested optional fields should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `instance_type` (string)
- `timestamp` (timestamp)

In addition to the required fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

WORK_FORCE Domain

Use the WORK_FORCE domain to forecast workforce demand. It supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types (p. 3).

Topics

Target Time Series Dataset Type

The following fields are required:

- `workforce_type` (string) – The type of work force labor being forecast. For example, call center demand or fulfillment center labor demand.
- `timestamp` (timestamp)
- `workforce_demand` (floating-point integer) – This is the `target` field for which Amazon Forecast generates a forecast.

The following dimension is optional and can be used to change forecasting granularity:

- `location` (string) – The location where the work force resources are sought. This should be used if you have multiple stores/locations.

Ideally, only these required fields and optional dimensions should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `workforce_type` (string)
- `timestamp` (timestamp)

In addition to the required fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

The following field is required:

- `workforce_type` (string)

The following fields are optional and might be useful in improving forecast results:

- `wages` (float) – The average wages for that particular workforce type.
- `shift_length` (string) – The length of the shift.
- `location` (string) – The location of the workforce.

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

WEB_TRAFFIC Domain

Use the WEB_TRAFFIC domain to forecast web traffic to a web property or a set of web properties. It supports the following dataset types. The relevant topics describe required and optional fields the dataset type supports. For information about how to map these fields to columns in your training data see Dataset Domains and Dataset Types (p. 3).

Topics

Target Time Series Dataset Type

The following fields are required:

- `item_id` (string) – A unique identifier for each web property being forecast.
- `timestamp` (timestamp)
- `value` (float) – This is the `target` field for which Amazon Forecast generates a forecast.

Ideally, only these required fields should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `item_id` (string)
- `timestamp` (timestamp)

In addition to the required fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

The following field is required:

- `item_id` (string)

The following field is optional and might be useful in improving forecast results:

- `category` (string)

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

METRICS Domain

Use the METRICS domain for forecasting metrics, such as revenue, sales, and cash flow. It supports the following dataset types. For each dataset type, we list required and optional fields. For information on how to map the fields to columns in your training data, see Dataset Domains and Dataset Types (p. 3).

Topics

Target Time Series Dataset Type

The following fields are required:

- `metric_name` (string)
- `timestamp` (timestamp)
- `metric_value` (floating-point integer) – This is the `target` field for which Amazon Forecast generates a forecast (for example, the amount of revenue generated on a particular day).

Ideally, only these required fields should be included. Other additional time series information should be included in a related time series dataset.

Related Time Series Dataset Type

The following fields are required:

- `metric_name` (string)
- `timestamp` (timestamp)

In addition to the required fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Item Metadata Dataset Type

The following field is required:

- `metric_name` (string)

The following field is optional and might be useful in improving forecast results:

- `category` (string)

In addition to the required and suggested optional fields, your training data can include other fields. To include other fields in the dataset, provide the fields in a schema when you create the dataset.

Choosing an Amazon Forecast Algorithm

Every Amazon Forecast predictor uses an algorithm to train a model, then uses the model to make a forecast using an input dataset group. To help you get started, Amazon Forecast provides the following predefined algorithms:

- Autoregressive Integrated Moving Average (ARIMA) Algorithm (p. 55)

  ```
  arn:aws:forecast:::algorithm/ARIMA
  ```
- DeepAR+ Algorithm (p. 56)*

  ```
  arn:aws:forecast:::algorithm/Deep_AR_Plus
  ```
- Exponential Smoothing (ETS) Algorithm (p. 62)

  ```
  arn:aws:forecast:::algorithm/ETS
  ```
- Non-Parametric Time Series (NPTS) Algorithm (p. 63)

  ```
  arn:aws:forecast:::algorithm/NPTS
  ```
- Prophet Algorithm (p. 66)

  ```
  arn:aws:forecast:::algorithm/Prophet
  ```

* Supports hyperparameter optimization (HPO)

Autoregressive Integrated Moving Average (ARIMA) Algorithm

Autoregressive Integrated Moving Average (ARIMA) is a commonly-used local statistical algorithm for time-series forecasting. ARIMA captures standard temporal structures (patterned organizations of time) in the input dataset. The Amazon Forecast ARIMA algorithm calls the Arima function in the `Package 'forecast'` of the Comprehensive R Archive Network (CRAN).

How ARIMA Works

The ARIMA algorithm is especially useful for datasets that can be mapped to stationary time series. The statistical properties of stationary time series, such as autocorrelations, are independent of time. Datasets with stationary time series usually contain a combination of signal and noise. The signal may exhibit a pattern of sinusoidal oscillation or have a seasonal component. ARIMA acts like a filter to separate the signal from the noise, and then extrapolates the signal in the future to make predictions.

ARIMA Hyperparameters and Tuning

For information about ARIMA hyperparameters and tuning, see the `Arima` function documentation in the Package 'forecast' of CRAN.

Amazon Forecast converts the `DataFrequency` parameter specified in the CreateDataset (p. 113) operation to the `frequency` parameter of the R ts function using the following table:

DataFrequency (string)	R ts frequency (integer)
Y	1
M	12
W	52
D	7
H	24
30min	2
15min	4
10min	6
5min	12
1min	60

For frequencies less than 24 or short time series, the hyperparameters are set using the `auto.arima` function of the `Package 'forecast'` of CRAN. For frequencies greater than or equal to 24 and long time series, we use a Fourier series with K = 4, as described here, Forecasting with long seasonal periods.

Supported data frequencies that aren't in the table default to a `ts` frequency of 1.

DeepAR+ Algorithm

Amazon Forecast DeepAR+ is a supervised learning algorithm for forecasting scalar (one-dimensional) time series using recurrent neural networks (RNNs). Classical forecasting methods, such as autoregressive integrated moving average (ARIMA) or exponential smoothing (ETS), fit a single model to each individual time series, and then use that model to extrapolate the time series into the future. In many applications, however, you have many similar time series across a set of cross-sectional units. These time-series groupings demand different products, server loads, and requests for web pages. In this case, it can be beneficial to train a single model jointly over all of the time series. DeepAR+ takes this approach. When your dataset contains hundreds of feature time series, the DeepAR+ algorithm outperforms the standard ARIMA and ETS methods. You can also use the trained model for generating forecasts for new time series that are similar to the ones it has been trained on.

Topics
- How DeepAR+ Works (p. 56)
- DeepAR+ Hyperparameters (p. 59)
- Tune DeepAR+ Models (p. 61)

How DeepAR+ Works

During training, DeepAR+ uses a training dataset and an optional testing dataset. It uses the testing dataset to evaluate the trained model. In general, the training and testing datasets don't have to contain

the same set of time series. You can use a model trained on a given training set to generate forecasts for the future of the time series in the training set, and for other time series. Both the training and the testing datasets consist of (preferably more than one) target time series. Optionally, they can be associated with a vector of feature time series and a vector of categorical features (for details, see DeepAR Input/Output Interface in the *Amazon SageMaker Developer Guide*). The following example shows how this works for an element of a training dataset indexed by i. The training dataset consists of a target time series, $z_{i,t}$, and two associated feature time series, $x_{i,1,t}$ and $x_{i,2,t}$.
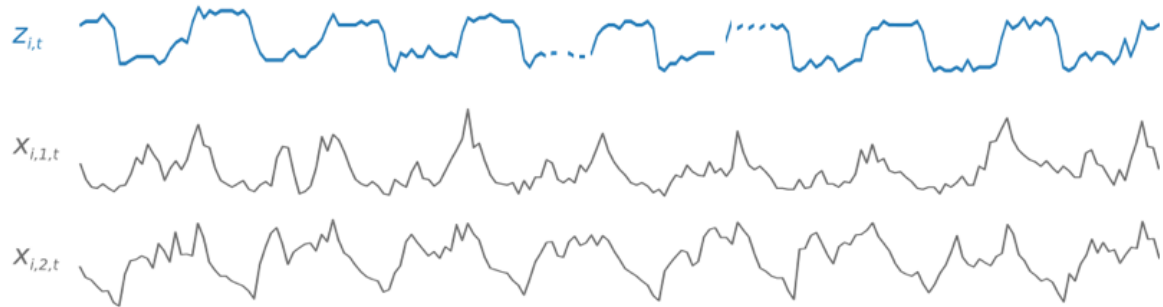
The target time series might contain missing values (denoted in the graphs by breaks in the time series). DeepAR+ supports only feature time series that are known in the future. This allows you to run counterfactual "what-if" scenarios. For example, "What happens if I change the price of a product in some way?"

Each target time series can also be associated with a number of categorical features. You can use these to encode that a time series belongs to certain groupings. Using categorical features allows the model to learn typical behavior for those groupings, which can increase accuracy. A model implements this by learning an embedding vector for each group that captures the common properties of all time series in the group.

To facilitate learning time-dependent patterns, such as spikes during weekends, DeepAR+ automatically creates feature time series based on time-series granularity. For example, DeepAR+ creates two feature time series (day of the month and day of the year) at a weekly time-series frequency. It uses these derived feature time series along with the custom feature time series that you provide during training and inference. The following example shows two derived time-series features: $u_{i,1,t}$ represents the hour of the day, and $u_{i,2,t}$ the day of the week.
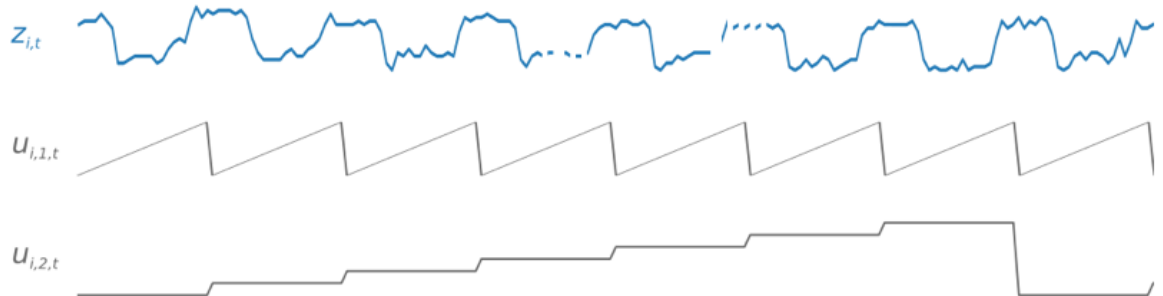
DeepAR+ automatically includes these feature time series based on the data frequency and the size of training data. The following table lists the features that can be derived for each supported basic time frequency.

Frequency of the Time Series	Derived Features
Minute	minute-of-hour, hour-of-day, day-of-week, day-of-month, day-of-year

Frequency of the Time Series	Derived Features
Hour	hour-of-day, day-of-week, day-of-month, day-of-year
Day	day-of-week, day-of-month, day-of-year
Week	day-of-month, week-of-year
Month	month-of-year

A DeepAR+ model is trained by randomly sampling several training examples from each of the time series in the training dataset. Each training example consists of a pair of adjacent context and prediction windows with fixed predefined lengths. The `context_length` hyperparameter controls how far in the past the network can see, and the `ForecastHorizon` parameter controls how far in the future predictions can be made. During training, Amazon Forecast ignores elements in the training dataset with time series shorter than the specified prediction length. The following example shows five samples, with a context length (highlighted in green) of 12 hours and a prediction length (highlighted in blue) of 6 hours, drawn from element i. For the sake of brevity, we've excluded the feature time series $x_{i,1,t}$ and $u_{i,2,t}$.
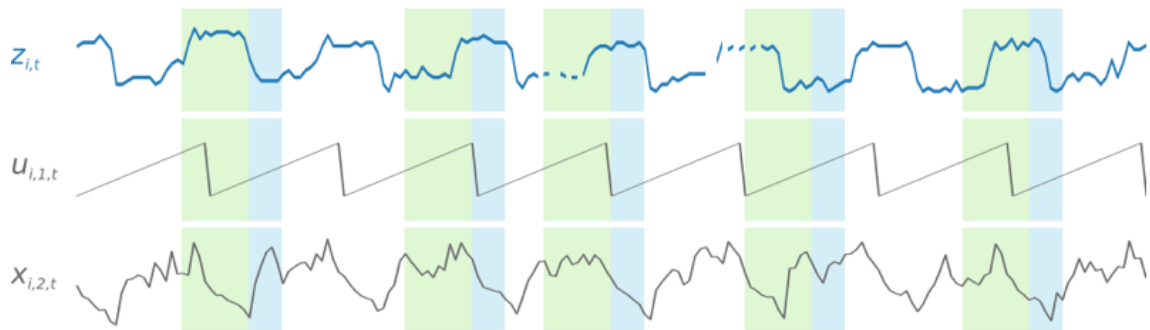
To capture seasonality patterns, DeepAR+ also automatically feeds lagged (past period) values from the target time series. In our example with samples taken at an hourly frequency, for each time index $t = T$, the model exposes the $z_{i,t}$ values which occurred approximately one, two, and three days in the past (highlighted in pink).
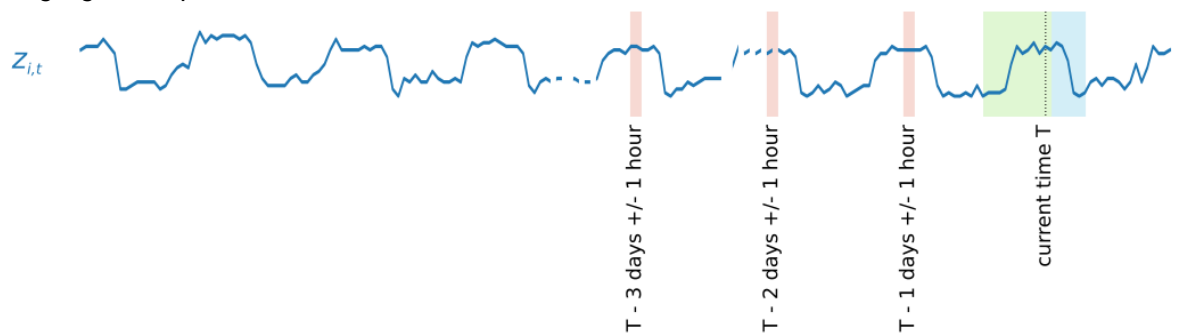
For inference, the trained model takes as input the target time series, which might or might not have been used during training, and forecasts a probability distribution for the next `ForecastHorizon` values. Because DeepAR+ is trained on the entire dataset, the forecast takes into account learned patterns from similar time series.

For information on the mathematics behind DeepAR+, see DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks on the Cornell University Library website.

DeepAR+ Hyperparameters

The following table lists the hyperparameters that you can use in the DeepAR+ algorithm. Parameters in bold participate in hyperparameter optimization (HPO).

Parameter Name	Description

`context_length`	The number of time points that the model reads in before making the prediction. The value for this parameter should be about the same as the `ForecastHorizon`. The model also receives lagged inputs from the target, so `context_length` 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.
	
	Valid values
	
	    Positive integers
	Typical values
	
	    ceil(0.1 * `ForecastHorizon`) to min(200, 10 * `ForecastHorizon`)
	Default value
	
	    2 * `ForecastHorizon`

`epochs`	The maximum number of passes to go over the training data. The optimal value depends on your data size and learning rate. Smaller datasets and lower learning rates both require more epochs, to achieve good results.
	
	Valid values
	
	    Positive integers
	Typical values
	
	    10 to 1000
	Default value
	
	    500

`learning_rate`	The learning rate used in training.
	
	Valid values
	
	    Positive floating-point numbers
	Typical values
	
	    0.0001 to 0.1
	Default value
	
	    0.001

`learning_rate_decay`	The rate at which the learning rate decreases. At most, the learning rate is reduced `max_learning_rate_decays` times, then training stops. This parameter will be used only if `max_learning_rate_decays` is greater than 0.

Parameter Name	Description
	Valid values
	Positive floating-point numbers
	Typical values
	0.5 to 0.8 (inclusive)
	Default value
	0.5
`likelihood`	The model generates a probabilistic forecast, and can provide quantiles of the distribution and return samples. Depending on your data, choose an appropriate likelihood (noise model) that is used for uncertainty estimates.
	Valid values
	`beta`: Use for real-valued targets between 0 and 1, inclusively. `deterministic-L1`: A loss function that does not estimate uncertainty and only learns a point forecast. `gaussian`: Use for real-valued data. `negative-binomial`: Use for count data (non-negative integers). `piecewise-linear`: Use for flexible distributions. `student-T`: Use this alternative for real-valued data for bursty data.
	Default value
	`student-T`
`max_learning_rate_decays`	The maximum number of learning rate reductions that should occur.
	Valid values
	Positive integers
	Typical values
	0 to 10
	Default value
	0
`num_averaged_models`	In DeepAR+, a training trajectory can encounter multiple models. Each model might have different forecasting strengths and weaknesses. DeepAR+ can average the model behaviors to take advantage of the strengths of all models.
	Valid values
	Positive integers
	Typical values
	1 to 5 (inclusive)
	Default value
	1

Parameter Name	Description

`num_cells`	The number of cells to use in each hidden layer of the RNN.
	
	Valid values
	
	  Positive integers
	Typical values
	
	  30 to 100
	Default value
	
	  40

`num_layers`	The number of hidden layers in the RNN.
	
	Valid values
	
	  Positive integers
	Typical values
	
	  1 to 4
	Default value
	
	  2

Tune DeepAR+ Models

To tune Amazon Forecast DeepAR+ models, follow these recommendations for optimizing the training process and hardware configuration.

Best Practices for Process Optimization

To achieve the best results, follow these recommendations:

- Except when splitting the training and testing datasets, always provide entire time series for training and testing, and when calling the model for inference. Regardless of how you set `context_length`, don't divide the time series or provide only a part of it. The model will use data points further back than `context_length` for the lagged values feature.

- For model tuning, you can split the dataset into training and testing datasets. In a typical evaluation scenario, you should test the model on the same time series used in training, but on the future `ForecastHorizon` time points immediately after the last time point visible during training. To create training and testing datasets that satisfy these criteria, use the entire dataset (all of the time series) as a testing dataset and remove the last `ForecastHorizon` points from each time series for training. This way, during training, the model doesn't see the target values for time points on which it is evaluated during testing. In the test phase, the last `ForecastHorizon` points of each time series in the testing dataset are withheld and a prediction is generated. The forecast is then compared with the actual values for the last `ForecastHorizon` points. You can create more complex evaluations by repeating time series multiple times in the testing dataset, but cutting them off at different end points. This produces accuracy metrics that are averaged over multiple forecasts from different time points.

- Avoid using very large values (> 400) for the `ForecastHorizon` because this slows down the model and makes it less accurate. If you want to forecast further into the future, consider aggregating to a higher frequency. For example, use `5min` instead of `1min`.

- Because of lags, the model can look further back than `context_length`. Therefore, you don't have to set this parameter to a large value. A good starting point for this parameter is the same value as the `ForecastHorizon`.
- Train DeepAR+ models with as many time series as are available. Although a DeepAR+ model trained on a single time series might already work well, standard forecasting methods such as ARIMA or ETS might be more accurate and are more tailored to this use case. DeepAR+ starts to outperform the standard methods when your dataset contains hundreds of feature time series. Currently, DeepAR+ requires that the total number of observations available, across all training time series, is at least 300.

Exponential Smoothing (ETS) Algorithm

Exponential Smoothing (ETS) is a commonly-used local statistical algorithm for time-series forecasting. The Amazon Forecast ETS algorithm calls the ets function in the `Package 'forecast'` of the Comprehensive R Archive Network (CRAN).

How ETS Works

The ETS algorithm is especially useful for datasets with seasonality and other prior assumptions about the data. ETS computes a weighted average over all observations in the input time series dataset as its prediction. The weights are exponentially decreasing over time, rather than the constant weights in simple moving average methods. The weights are dependent on a constant parameter, which is known as the smoothing parameter.

ETS Hyperparameters and Tuning

For information about ETS hyperparameters and tuning, see the ets function documentation in the Package 'forecast' of CRAN.

Amazon Forecast converts the `DataFrequency` parameter specified in the CreateDataset (p. 113) operation to the `frequency` parameter of the R ts function using the following table:

DataFrequency (string)	R ts frequency (integer)
Y	1
M	12
W	52
D	7
H	24
30min	2
15min	4
10min	6
5min	12
1min	60

Supported data frequencies that aren't in the table default to a `ts` frequency of 1.

Non-Parametric Time Series (NPTS) Algorithm

The Amazon Forecast Non-Parametric Time Series (NPTS) algorithm is a scalable, probabilistic baseline forecaster. It predicts the future value distribution of a given time series by sampling from past observations. The predictions are bounded by the observed values. NPTS is especially useful when the time series is intermittent (or sparse, containing many 0s) and bursty. For example, forecasting demand for individual items where the time series has many low counts. Amazon Forecast provides variants of NPTS that differ in which of the past observations are sampled and how they are sampled. To use an NPTS variant, you choose a hyperparameter setting.

How NPTS Works

Similar to classical forecasting methods, such as exponential smoothing (ETS) and autoregressive integrated moving average (ARIMA), NPTS generates predictions for each time series individually. The time series in the dataset can have different lengths. The time points where the observations are available are called the training range and the time points where the prediction is desired are called the prediction range.

Amazon Forecast NPTS forecasters have the following variants: NPTS, seasonal NPTS, climatological forecaster, and seasonal climatological forecaster.

Topics

- NPTS (p. 63)
- Seasonal NPTS (p. 63)
- Climatological Forecaster (p. 64)
- Seasonal Climatological Forecaster (p. 64)
- Seasonal Features (p. 64)
- Best Practices (p. 64)

NPTS

In this variant, predictions are generated by sampling from all observations in the training range of the time series. However, instead of uniformly sampling from all of the observations, this variant assigns weight to each of the past observations according to how far it is from the current time step where the prediction is needed. In particular, it uses weights that decay exponentially according to the distance of the past observations. In this way, the observations from the recent past are sampled with much higher probability than the observations from the distant past. This assumes that the near past is more indicative for the future than the distant past. You can control the amount of decay in the weights with the `exp_kernel_weights` hyperparameter.

To use this NPTS variant in Amazon Forecast, set the `use_seasonal_model` hyperparameter to `False` and accept all other default settings.

Seasonal NPTS

The seasonal NPTS variant is similar to NPTS except that instead of sampling from all of the observations, it uses only the observations from the past *seasons* . By default, the season is determined by the granularity of the time series. For example, for an hourly time series, to predict for hour t, this variant samples from the observations corresponding to the hour t on the previous days. Similar to NPTS, observation at hour t on the previous day is given more weight than the observations at hour t on earlier days. For more information about how to determine seasonality based on the granularity of the time series, see the section called "Seasonal Features" (p. 64).

If you provide time series features with the `feat_dynamic_real` hyperparameter, seasonality is determined by both the granularity and the `feat_dynamic_real` hyperparameter. To use only the `feat_dynamic_real` hyperparameter to define seasonality, set the `use_default_time_features` hyperparameter to `False`. The `feat_dynamic_real` hyperparameter is turned on in Amazon Forecast by passing in the related time-series CSV file.

Climatological Forecaster

The climatological forecaster variant samples all of the past observations with uniform probability.

To use the climatological forecaster, set the `kernel_type` hyperparameter to `uniform` and the `use_seasonal_model` hyperparameter to `False`. Accept the default settings for all other hyperparameters.

Seasonal Climatological Forecaster

Similar to seasonal NPTS, the seasonal climatological forecaster samples the observations from past seasons, but samples them with uniform probability.

To use the seasonal climatological forecaster, set the `kernel_type` hyperparameter to `uniform`. Accept all other default settings for all of the other hyperparameters.

Seasonal Features

To determine what corresponds to a season for the seasonal NPTS and seasonal climatological forecaster, use the features listed in the following table. The table lists the derived features for the supported basic time frequencies, based on granularity. Amazon Forecast includes these feature time series, so you don't have to provide them.

Frequency of the Time Series	Feature to Determine Seasonality
Minute	minute-of-hour
Hour	hour-of-day
Day	day-of-week
Week	day-of-month
Month	month-of-year

Best Practices

When using the Amazon Forecast NPTS algorithms, consider the following best practices for preparing the data and achieving optimal results:

- Because NPTS generates predictions for each time series individually, provide the entire time series when calling the model for prediction. Also, accept the default value of the `context_length` hyperparameter. This causes the algorithm to use the entire time series. If you change the `context_length` (because the training data is too long), make sure it is large enough and covers multiple past seasons. For example, for a daily time series, this value must be at least 365 days (provided that you have that amount of data).
- If the data has seasonality patterns, the seasonal NPTS algorithm typically works better. If external events, such as special holidays and promotions, have an effect on the time series, then provide those

features in the `feat_dynamic_real` hyperparameter and use seasonal NPTS. In this case, you must also provide the `feat_dynamic_real` hyperparameter for both training and prediction ranges by providing the related time-series CSV file to Amazon Forecast.

NPTS Hyperparameters

The following table lists the hyperparameters that you can use in the NPTS algorithm.

Parameter Name	Description

context_length	The number of time-points in the past that the model uses for making the prediction. By default, it uses all of the time points in the training range. Typically, the value for this hyperparameter should be large and should cover multiple past seasons. For example, for the daily time series this value must be at least 365 days.
	
	Valid values
	
	    Positive integers
	Default value
	
	    The length of the training time series

kernel_type	The kernel to use to define the weights used for sampling past observations.
	
	Valid values
	
	    `exponential` or `uniform`
	Default values
	
	    `exponential`

exp_kernel_weights	Valid only when `kernel_type` is `exponential`.
	
	The scaling parameter of the kernel. For faster (exponential) decay in the weights given to the observations in the distant past, use a large value.
	
	Valid values
	
	    Positive floating-point numbers
	Default value
	
	    `0.01`

use_seasonal_model	Whether to use a seasonal variant.
	
	Valid values
	
	    `True` or `False`
	Default value
	
	    `True`

use_default_time_features	Valid only for the *seasonal NPTS* and *seasonal climatological forecaster* variants. Whether to use seasonal features based on the granularity of the time series to determine seasonality.

Parameter Name	Description

	Valid values
	
	    `True` or `False`
	Default value
	
	    `True`

Prophet Algorithm

Prophet is a popular local Bayesian structural time series model. The Amazon Forecast Prophet algorithm uses the Prophet class of the Python implementation of Prophet.

How Prophet Works

Prophet is especially useful for datasets that:

- Contain an extended time period (months or years) of detailed historical observations (hourly, daily, or weekly)
- Have multiple strong seasonalities
- Include previously known important, but irregular, events
- Have missing data points or large outliers
- Have non-linear growth trends that are approaching a limit

Prophet is an additive regression model with a piecewise linear or logistic growth curve trend. It includes a yearly seasonal component modeled using Fourier series and a weekly seasonal component modeled using dummy variables.

For more information, see Prophet: forecasting at scale.

Prophet Hyperparameters and Related Time Series

Amazon Forecast uses the default Prophet hyperparameters. Prophet also supports related time-series as features, provided to Amazon Forecast in the related time-series CSV file.

Evaluating Predictor Accuracy

To evaluate the accuracy of an algorithm for various forecasting scenarios and to tune the predictor, use predictor metrics. Amazon Forecast uses backtesting to produce metrics.

Forecast automatically splits your input data into two datasets, training and test, as shown in the following figure. Forecast decides how to split the input data by using the `BackTestWindowOffset` parameter that you specify in the CreatePredictor (p. 129) operation, or if not specified, it uses the default value of the `ForecastHorizon` parameter. For more information, see EvaluationParameters (p. 206).
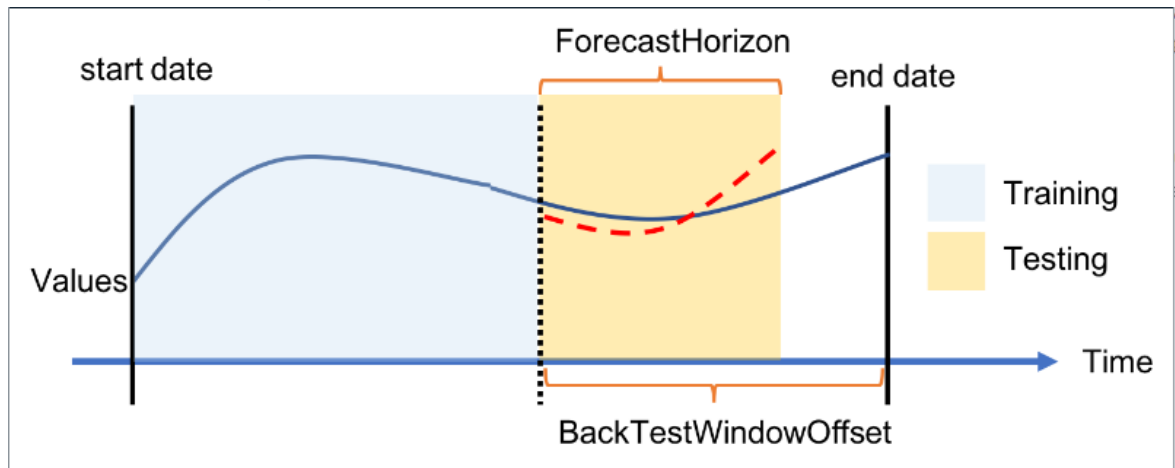
To evaluate the metrics in multiple backtest scenarios with different virtual forecast start dates, as shown in the following figure, use the `NumberOfBacktestWindows` parameter in the `CreatePredictor` operation. The default for the `NumberOfBacktestWindows` parameter is 1. If you use the default, Forecast uses the simple splitting method shown in the preceding figure.
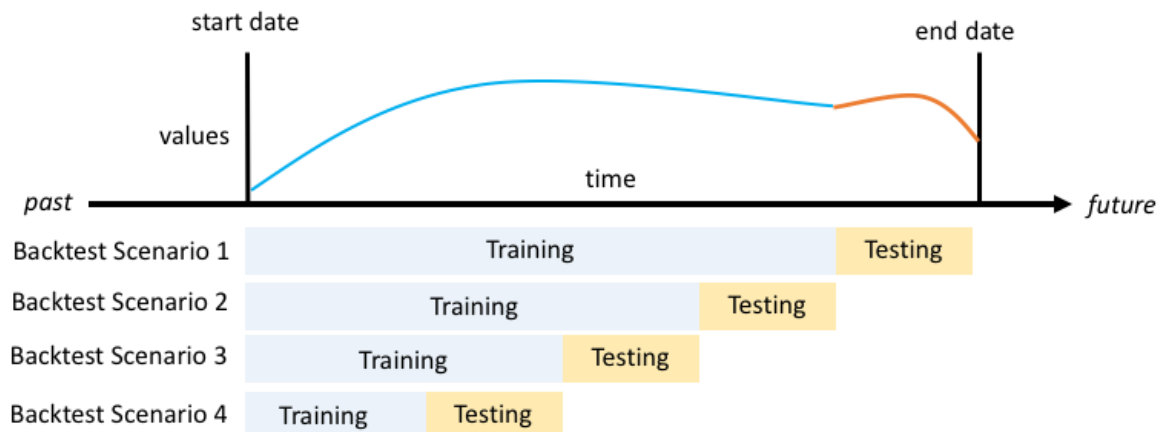
After training, Amazon Forecast calculates the root mean square error (RMSE) and weighted quantile losses to determine how well the model predicted the test data in each backtest window and the average value over all the backtest windows. These metrics measure the difference between the values predicted by the model and the actual values in the test dataset. To retrieve the metrics, you use the GetAccuracyMetrics (p. 170) operation.

Root Mean Square Error

RMSE is the square of the error term, which is the difference between the actual target value, $y_{i,t}$, and the predicted (forecasted) value, $\hat{y}_{i,t}$, where `i` denotes the item index ranging from 1 to the total number of items, `n`, and `t` denotes the time index of the time series ranging from 1 to the final time in the evaluation period, `T`.

$$\text{RMSE} = \sqrt{\frac{1}{nT}\sum_{i,t}(\hat{y}_{i,t} - y_{i,t})^2},$$

$$i = 1, \ldots, n$$

$$t = 1, \ldots T$$

The RMSE metric favors a model whose individual errors are of consistent magnitude because large variations in error increase the RMSE. Because of the squared error, a few poorly predicted values in an otherwise good forecast can increase the RMSE.

Prediction Quantiles and MAPE

Prediction quantiles (intervals) express the uncertainty in the forecasts. By calculating prediction quantiles, the model shows how much uncertainty is associated with each forecast. Without accompanying prediction quantiles, point forecasts have limited value.

Predicting forecasts at different quantiles is particularly useful when the costs of under and over predicting differ. Amazon Forecast provides probabilistic predictions at three distinct quantiles—10%, 50%, and 90%—and calculates the associated loss (error) at each quantile. The *weighted quantile loss* (wQuantileLoss) calculates how far off the forecast is from actual demand in either direction. This is calculated as a percentage of demand on average in each quantile. This metric helps capture the bias inherent in each quantile, which can't be captured by a calculation like MAPE (Mean Absolute Percentage Error), where the weights are equal. As with MAPE and RMSE, lower wQuantileLoss errors indicate better overall forecast accuracy.

The weighted quantile loss is calculated as follows:

$$\text{wQuantileLoss}[\tau] = 2\frac{\sum_{i,t}[\tau \max(y_{i,t} - q_{i,t}^{(\tau)}, 0) + (1 - \tau)\max(q_{i,t}^{(\tau)} - y_{i,t}, 0)]}{\sum_{i,t}|y_{i,t}|}$$

$q_{i,t}^{(\tau)}$ is the τ-quantile that the model predicts. τ is in the set {0.1, 0.2, ..., 0.9}.

Amazon Forecast calculates the weighted P10, P50, and P90 quantile losses, where τ is in the set {0.1, 0.5, 0.9}, respectively. This covers the standard 80% confidence interval. For RMSE, Amazon Forecast uses the P50 forecast to represent the predicted value, for example, $\hat{y}_{i,t}$ = $q_{i,t}^{(0.5)}$.

When the sum of the exact target over all items and all time is approximately zero in a given backtest window, the weighted quantile loss expression is undefined. In this case, Amazon Forecast outputs the unweighted quantile loss, which is the numerator in the above wQuantileLoss expression.

wQuantileLoss[0.1]: For the P10 prediction, the true value is expected to be lower than the predicted value 10% of the time.

For example, suppose that you're a retailer and you want to forecast product demand for winter gloves that sell well only during the fall and winter. If you don't have a lot of storage space and the cost of invested capital is high, or if the price of being overstocked on winter gloves concerns you, you might use the P10 quantile to order a relatively low number of winter gloves. You know that the P10 forecast overestimates the demand for your winter gloves only 10% of the time, so 90% of the time you'll be sold out of your winter gloves.

wQuantileLoss[0.5]: For the P50 prediction, the true value is expected to be lower than the predicted value 50% of the time. In most cases, the point forecasts that you generate internally or with

other forecasting tools should match the P50 forecasts. If τ = 0.5, both weights are equal and the wQuantileLoss[0.5] reduces to the commonly used Mean Absolute Percentage Error (MAPE):

$$\mathrm{MAPE} = \frac{\sum_{i,t} |\hat{y}_{i,t} - y_{i,t}|}{\sum_{i,t} |y_{i,t}|}$$

where $\hat{y}_{i,t} = q_{i,t}{}^{(0.5)}$.

Forecast uses the scaling factor of 2 in the wQuantileLoss formula to cancel the 0.5 factor to obtain the exact MAPE expression.

Continuing the winter gloves example, if you know that there'll be a moderate amount of demand for the gloves and aren't concerned about being overstocked, you might choose to use the P50 quantile to order gloves.

wQuantileLoss[0.9]: For the P90 prediction, the true value is expected to be lower than the predicted value 90% of the time.

If you determine that being understocked on gloves will result in huge amounts of lost revenue—for example, the cost of not selling gloves is extremely high or the cost of invested capital is low—you might choose to use the P90 quantile to order gloves.

The following figure of a forecast that has a Gaussian distribution, shows the quantiles that divide the forecast into four regions of equal probability. For information about the quantiles of a distribution, see Quantile on Wikipedia.
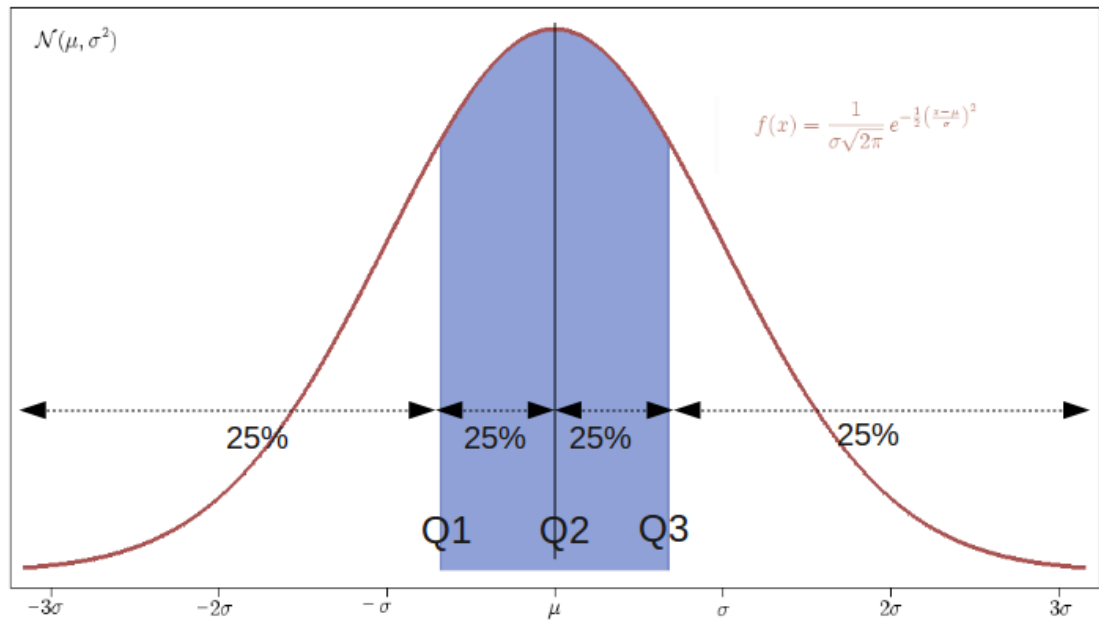
Updating Data

As you collect new data, you may want to use it to generate new forecasts. Forecast does not automatically retrain a predictor when you import an updated dataset, but you can use an existing predictor to generate forecasts with the updated data. For instance, if you collect daily sales data and want to include new datapoints in your forecast, you could import the updated data and use it to generate a forecast without training a new predictor. If, however, you want your predictor to be trained off of the new data, you must create a new predictor.

To generate a forecast off of new data:

1. Upload the updated CSV file to an Amazon S3 bucket. The updated CSV should still contain all of your existing data.
2. Create a dataset import job with the new data. The most recent import job is the one that forecasts are generated off of.
3. Create a new forecast using the existing predictor.
4. Retreieve the forecast as usual.

Security in Amazon Forecast

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from data centers and network architectures that are built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to Amazon Forecast, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using Forecast. The following topics show you how to configure Forecast to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Forecast resources.

Topics

Data Protection in Amazon Forecast

Amazon Forecast conforms to the AWS shared responsibility model, which includes regulations and guidelines for data protection. AWS is responsible for protecting the global infrastructure that runs all the AWS services. AWS maintains control over data hosted on this infrastructure, including the security configuration controls for handling customer content and personal data. AWS customers and APN partners, acting either as data controllers or data processors, are responsible for any personal data that they put in the AWS Cloud.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM), so that each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources.

- Set up API and user activity logging with AWS CloudTrail.

- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon Simple Storage Service (Amazon S3).

We strongly recommend that you never put sensitive identifying information, such as your customers' account numbers, into free-form fields such as a **Name** field. This includes when you work with Forecast or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into Forecast or other services might get picked up for inclusion in diagnostic logs. When you provide a URL to an external server, don't include credentials information in the URL to validate your request to that server.

For more information about data protection, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

Encryption at Rest

Use one of your own AWS Key Management Service (AWS KMS) symmetric keys to encrypt Forecast data in your Amazon S3 buckets.

Encryption in Transit

Amazon Forecast uses TLS 1.2 with AWS certificates to encrypt data sent to other AWS services.

Key Management

The default Amazon S3 keys are managed by AWS. It is the responsibility of the customer to manage any customer-provided AWS Key Management Service (AWS KMS) keys.

Identity and Access Management for Amazon Forecast

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use Forecast resources. IAM is an AWS service that you can use with no additional charge.

Topics

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work you do in Forecast.

Service user – If you use the Forecast service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Forecast features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in Forecast, see Troubleshooting Amazon Forecast Identity and Access (p. 82).

Service administrator – If you're in charge of Forecast resources at your company, you probably have full access to Forecast. It's your job to determine which Forecast features and resources your employees should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with Forecast, see How Amazon Forecast Works with IAM (p. 76).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Forecast. To view example Forecast identity-based policies that you can use in IAM, see Amazon Forecast Identity-Based Policy Examples (p. 78).

Authenticating With Identities

Authentication is how you sign in to AWS using your identity credentials. For more information about signing in using the AWS Management Console, see The IAM Console and Sign-in Page in the *IAM User Guide*.

You must be *authenticated* (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role. You can also use your company's single sign-on authentication, or even sign in using Google or Facebook. In these cases, your administrator previously set up identity federation using IAM roles. When you access AWS using credentials from another company, you are assuming a role indirectly.

To sign in directly to the AWS Management Console, use your password with your root user email or your IAM user name. You can access AWS programmatically using your root user or IAM user access keys. AWS provides SDK and command line tools to cryptographically sign your request using your credentials. If you don't use AWS tools, you must sign the request yourself. Do this using *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see Signature Version 4 Signing Process in the *AWS General Reference*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Using Multi-Factor Authentication (MFA) in AWS in the *IAM User Guide*.

AWS Account Root User

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

IAM Users and Groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. An IAM user can have long-term credentials such as a user name and password or a set of access keys. To learn how to generate access keys, see Managing Access Keys for IAM Users in the *IAM User Guide*. When you generate access keys for an IAM user, make sure you view and securely save the key

pair. You cannot recover the secret access key in the future. Instead, you must generate a new access key pair.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see When to Create an IAM User (Instead of a Role) in the *IAM User Guide*.

IAM Roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM Roles in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an identity provider. For more information about federated users, see Federated Users and Roles in the *IAM User Guide*.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see How IAM Roles Differ from Resource-based Policies in the *IAM User Guide*.
- **AWS service access** – A service role is an IAM role that a service assumes to perform actions in your account on your behalf. When you set up some AWS service environments, you must define a role for the service to assume. This service role must include all the permissions that are required for the service to access the AWS resources that it needs. Service roles vary from service to service, but many allow you to choose your permissions as long as you meet the documented requirements for that service. Service roles provide access only within your account and cannot be used to grant access to services in other accounts. You can create, modify, and delete a service role from within IAM. For example, you can create a role that allows Amazon Redshift to access an Amazon S3 bucket on your behalf and then load data from that bucket into an Amazon Redshift cluster. For more information, see Creating a Role to Delegate Permissions to an AWS Service in the *IAM User Guide*.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances in the *IAM User Guide*.

To learn whether to use IAM roles, see When to Create an IAM Role (Instead of a User) in the *IAM User Guide*.

Managing Access Using Policies

You control access in AWS by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when an entity (root user, IAM user, or IAM role) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON Policies in the *IAM User Guide*.

An IAM administrator can use policies to specify who has access to AWS resources, and what actions they can perform on those resources. Every IAM entity (user or role) starts with no permissions. In other words, by default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-Based Policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, role, or group. These policies control what actions that identity can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM Policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choosing Between Managed Policies and Inline Policies in the *IAM User Guide*.

Resource-Based Policies

Resource-based policies are JSON policy documents that you attach to a resource such as an Amazon S3 bucket. Service administrators can use these policies to define what actions a specified principal (account member, user, or role) can perform on that resource and under what conditions. Resource-based policies are inline policies. There are no managed resource-based policies.

Access Control Lists (ACLs)

Access control lists (ACLs) are a type of policy that controls which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format. Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access Control List (ACL) Overview in the *Amazon Simple Storage Service Developer Guide*.

Other Policy Types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of

entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions Boundaries for IAM Entities in the *IAM User Guide*.

- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs Work in the *AWS Organizations User Guide*.

- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session Policies in the *IAM User Guide*.

Multiple Policy Types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy Evaluation Logic in the *IAM User Guide*.

How Amazon Forecast Works with IAM

Before you use IAM to manage access to Amazon Forecast, you should understand what IAM features are available to use with Forecast. To get a high-level view of how Forecast and other AWS services work with IAM, see AWS Services That Work with IAM in the *IAM User Guide*.

Topics

Forecast Identity-Based Policies

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. Forecast supports specific actions, resources, and condition keys. To learn about all of the elements that you use in a JSON policy, see IAM JSON Policy Elements Reference in the *IAM User Guide*.

Actions

The `Action` element of an IAM identity-based policy describes the specific action or actions that will be allowed or denied by the policy. Policy actions usually have the same name as the associated AWS API operation. The action is used in a policy to grant permissions to perform the associated operation.

Policy actions in Forecast use the following prefix before the action: `forecast:`. For example, to grant someone permission to run an FOR dataset group creation job with the `CreateDatasetGroup` API operation, you include the `forecast:CreateDatasetGroup` action in their policy. Policy statements must include either an `Action` or `NotAction` element. Forecast defines its own set of actions that describe tasks that you can perform with this service.

To specify multiple actions in a single statement, separate them with commas as follows.

```
"Action": [
      "forecast:action1",
      "forecast:action2"
```

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word `Describe`, include the following action.

```
"Action": "forecast:Describe*"
```

To see a list of Forecast actions, see Actions Defined by Amazon Forecast in the *IAM User Guide*.

Resources

The `Resource` element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. You specify a resource using an ARN or using the wildcard (*) to indicate that the statement applies to all resources.

An Amazon Forecast dataset resource has the following ARN.

```
arn:${Partition}:forecast:${Region}:${Account}:dataset/${DatasetName}
```

For more information about the format of ARNs, see Amazon Resource Names (ARNs) and AWS Service Namespaces.

For example, to specify the dataset called `MyDataset` in your statement, use the following ARN.

```
"Resource": "arn:aws:forecast:us-east-1:123456789012:dataset/MyDataset"
```

To specify all datasets that belong to a specific account, use the wildcard (*).

```
"Resource": "arn:aws:forecast:us-east-1:123456789012:dataset/*"
```

Some Forecast actions, such as those for creating resources, cannot be performed on a specific resource. In those cases, you must use the wildcard (*).

```
"Resource": "*"
```

To see a list of Forecast resource types and their ARNs, see Resources Defined by Amazon Forecast in the *IAM User Guide*. To learn with which actions you can specify the ARN of each resource, see Actions Defined by Amazon Forecast.

Condition Keys

Forecast does not provide any service-specific condition keys.

Examples

To view examples of Forecast identity-based policies, see Amazon Forecast Identity-Based Policy Examples (p. 78).

Forecast Resource-Based Policies

Forecast does not support resource-based policies.

Authorization Based on Forecast Tags

Forecast does not support tagging resources or controlling access based on tags.

Forecast IAM Roles

An IAM role is an entity within your AWS account that has specific permissions.

Using Temporary Credentials with Forecast

You can use temporary credentials to sign in with federation, assume an IAM role, or to assume a cross-account role. You obtain temporary security credentials by calling AWS STS API operations such as AssumeRole or GetFederationToken.

Forecast supports using temporary credentials.

Service-Linked Roles

Service-linked roles allow AWS services to access resources in other services to complete an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view but not edit the permissions for service-linked roles.

Forecast does not support service-linked roles.

Service Roles

This feature allows a service to assume a service role on your behalf. This role allows the service to access resources in other services to complete an action on your behalf. Service roles appear in your IAM account and are owned by the account. This means that an IAM administrator can change the permissions for this role. However, doing so might break the functionality of the service.

Forecast supports service roles.

Amazon Forecast Identity-Based Policy Examples

By default, IAM users and roles don't have permission to create or modify Forecast resources. They also can't perform tasks using the AWS Management Console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the specified resources they need. The administrator must then attach those policies to the IAM users or groups that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see Creating Policies on the JSON Tab in the *IAM User Guide*.

Topics

Policy Best Practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete Forecast resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get Started Using AWS Managed Policies** – To start using Forecast quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see Get Started Using Permissions With AWS Managed Policies in the *IAM User Guide*.
- **Grant Least Privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later. For more information, see Grant Least Privilege in the *IAM User Guide*.
- **Enable MFA for Sensitive Operations** – For extra security, require IAM users to use multi-factor authentication (MFA) to access sensitive resources or API operations. For more information, see Using Multi-Factor Authentication (MFA) in AWS in the *IAM User Guide*.
- **Use Policy Conditions for Extra Security** – To the extent that it's practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions to allow requests only within a specified date or time range, or to require the use of SSL or MFA. For more information, see IAM JSON Policy Elements: Condition in the *IAM User Guide*.

Using the Forecast Console

To access the Amazon Forecast console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the Forecast resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (IAM users or roles) with that policy.

To ensure that those entities can still use the Forecast console, also attach the following AWS managed policy to the entities. For more information, see Adding Permissions to a User in the *IAM User Guide*:

```
AWSForecastFullAccess
```

The following policy grants full access to all Amazon Forecast actions when using the console:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "forecast:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "forecast.amazonaws.com"
                }
```

```
                }
            }
        ]
}
```

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that you're trying to perform.

Allow Users to View Their Own Permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                  "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS Managed (Predefined) Policies for Amazon Forecast

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. These AWS managed policies grant necessary permissions for common use cases so that you can avoid having to investigate which permissions are needed. For more information, see AWS Managed Policies in the *IAM User Guide*.

The following AWS managed policies, which you can attach to users in your account, are specific to Amazon Forecast:

- **AmazonForecastFullAccess** – Grants full access to Amazon Forecast resources and all of the supported operations.

You can review these permissions policies by signing in to the IAM console and searching for them.

You can also create your own custom IAM policies to allow permissions for Amazon Forecast actions and resources. You can attach these custom policies to the IAM users or groups that require them.

Customer Managed Policy Examples

In this section, you can find example user policies that grant permissions for various Amazon Forecast actions. These policies work when you are using the AWS SDKs or the AWS CLI. When you are using the console, see Using the Forecast Console (p. 79).

Examples

Example 1: Grant Account Administrator Permissions

After you set up an account (see Sign Up for AWS (p. 14)), you create an administrator user to manage your account. The administrator user can create users and manage their permissions.

To grant the administrator user all of the permissions available for your account, attach the following permissions policy to that user:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

Example 2: Allow All Amazon Forecast Actions

You might choose to create a user who has permissions for all Amazon Forecast actions but not for any of your other services (think of this user as a service-specific administrator). Attach the following permissions policy to this user:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "forecast:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
```

```
            "StringEquals": {
                "iam:PassedToService": "forecast.amazonaws.com"
            }
        }
    }
    ]
}
```

Example 3: Action-based Policy: Amazon Forecast Read-Only Access

The following policy grants permissions to Amazon Forecast actions that allow a user to list and describe resources:

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "forecast:DescribeDataset",
          "forecast:DescribeDatasetGroup",
          "forecast:DescribeDatasetImportJob",
          "forecast:DescribeForecast",
          "forecast:DescribeForecastExportJob",
          "forecast:DescribePredictor",
          "forecast:ListDatasetGroups",
          "forecast:ListDatasetImportJobs",
          "forecast:ListDatasets",
          "forecast:ListDatasetExportJobs",
          "forecast:ListForecasts",
          "forecast:ListPredictors"
        ],
        "Resource": "*"
      }
    ]
}
```

Troubleshooting Amazon Forecast Identity and Access

Use the following information to help you diagnose and fix common issues that you might encounter when working with Forecast and IAM.

Topics

I Am Not Authorized to Perform an Action in Forecast

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a *widget* but does not have `forecast:GetWidget` permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 forecast:GetWidget on resource: my-example-widget
```

In this case, Mateo asks his administrator to update his policies to allow him to access the *my-example-widget* resource using the `forecast:GetWidget` action.

I Am Not Authorized to Perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password. Ask that person to update your policies to allow you to pass a role to Forecast.

Some AWS services allow you to pass an existing role to that service, instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in Forecast. However, the action requires the service to have permissions granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform: iam:PassRole
```

In this case, Mary asks her administrator to update her policies to allow her to perform the `iam:PassRole` action.

I Want to View My Access Keys

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair.

Access keys consist of two parts: an access key ID (for example, `AKIAIOSFODNN7EXAMPLE`) and a secret access key (for example, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Like a user name and password, you must use both the access key ID and secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

> **Important**
> Do not provide your access keys to a third party, even to help find your canonical user ID. By doing this, you might give someone permanent access to your account.

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must add new access keys to your IAM user. You can have a maximum of two access keys. If you already have two, you must delete one key pair before creating a new one. To view instructions, see Managing Access Keys in the *IAM User Guide*.

I'm an Administrator and Want to Allow Others to Access Forecast

To allow others to access Forecast, you must create an IAM entity (user or role) for the person or application that needs access. They will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in Forecast.

To get started right away, see Creating Your First IAM Delegated User and Group in the *IAM User Guide*.

I Want to Allow People Outside of My AWS Account to Access My Forecast Resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether Forecast supports these features, see How Amazon Forecast Works with IAM (p. 76).
- To learn how to provide access to your resources across AWS accounts that you own, see Providing Access to an IAM User in Another AWS Account That You Own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing Access to AWS Accounts Owned by Third Parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing Access to Externally Authenticated Users (Identity Federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see How IAM Roles Differ from Resource-based Policies in the *IAM User Guide*.

Logging and Monitoring in Amazon Forecast

Monitoring is an important part of maintaining the reliability, availability, and performance of your Amazon Forecast applications. To monitor Amazon Forecast API calls, you can use AWS CloudTrail. To monitor the status of your Forecast assets and processes, use Amazon CloudWatch.

Topics
- Logging Forecast API Calls with AWS CloudTrail (p. 84)
- CloudWatch Metrics for Amazon Forecast (p. 86)

Logging Forecast API Calls with AWS CloudTrail

Amazon Forecast is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in Forecast. CloudTrail captures all API calls for Forecast as events. The calls captured include calls from the Forecast console and code calls to the Forecast API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon Simple Storage Service (Amazon S3) bucket, including events for Forecast. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to Forecast, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

Forecast Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Forecast, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for Forecast, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
- Receiving CloudTrail Log Files from Multiple Regions and Receiving CloudTrail Log Files from Multiple Accounts

All Forecast actions are logged by CloudTrail and are documented in the Amazon Forecast Developer Guide. For example, calls to the `CreateDataset` and `CreateForecast` actions generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity Element.

Understanding Forecast Log File Entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the `CreateDataset` action.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAIQ4PAJSMEEPNEXAMPLE",
    "arn": "arn:aws:iam::acct-id:user/userxyz",
    "accountId": "111111111111",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "userxyz"
  },
  "eventTime": "2018-11-21T23:53:06Z",
  "eventSource": "forecast.amazonaws.com",
  "eventName": "CreateDataset",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.168.0.1",
  "userAgent": "Boto3/1.7.82 Python/3.6.5 Linux/4.14.72-68.55.amzn1.x86_64
 Botocore/1.10.84",
  "requestParameters": {
    "domain": "CUSTOM",
```

```
          "datasetType": "TARGET_TIME_SERIES",
          "dataFormat": "CSV",
          "datasetName": "forecast_test_script_ds",
          "dataFrequency": "D",
          "timeStampFormat": "yyyy-MM-dd",
          "schema": {
            "attributes": [
              {
                "attributeName": "item_id",
                "attributeType": "string"
              },
              {
                "attributeName": "timestamp",
                "attributeType": "timestamp"
              },
              {
                "attributeName": "target_value",
                "attributeType": "float"
              },
              {
                "attributeName": "visits",
                "attributeType": "float"
              },
              {
                "attributeName": "was_open",
                "attributeType": "float"
              },
              {
                "attributeName": "promotion_applied",
                "attributeType": "float"
              }
            ]
          }
        },
        "responseElements": {
          "datasetName": "forecast_test_script_ds",
          "datasetArn": "arn:aws:forecast:us-west-2:acct-id:ds/forecast_test_script_ds"
        },
        "requestID": "EXAMPLE8-90ab-cdef-fedc-ba987EXAMPLE",
        "eventID": "EXAMPLE8-90ab-cdef-fedc-ba987EXAMPLE",
        "eventType": "AwsApiCall",
        "recipientAccountId": "111111111111"
}
```

CloudWatch Metrics for Amazon Forecast

This section contains information about the Amazon CloudWatch metrics available for Amazon Forecast.

The following table lists the Amazon Forecast metrics.

Metric	Dimension	Unit	Statistics	Description
DatasetSize		Kilobytes	Average, Sum, Min, Max	The total size of the datasets imported by Amazon Forecast into the customer's account.
DatasetSize	DatasetArn DatasetImportJobArn	Kilobytes	Average, Sum	The size of the dataset imported by the CreateDatasetImportJob (p. 120) operation.
CreatePredictorExecutionTime	PredictorArn	Seconds	Average, Sum	The time taken for training, inference, and metrics for a specific predictor. Amazon

Metric	Dimension	Unit	Statistics	Description
				Forecast normalizes the compute costs to a c5.xlarge instance to arrive at the number of hours consumed by the training job.
CreateForecastExecutionTime	ExecutionArn	Seconds	Average, Sum	The time taken for training and inference during forecast generation. Amazon Forecast normalizes the compute costs to a c5.xlarge instance to arrive at the number of hours consumed by the training job.
TimeSeriesForecastsGenerated		Count	Average, Sum, Min, Max	The number of unique time series forecasts generated for each quantile across all predictors in the account. Forecasts are billed to the nearest 1000 and charged on a per 1,000 basis.
TimeSeriesForecastsGenerated	PredictorArn	Count	Average, Sum, Min, Max	The number of unique time series forecasts generated for each quantile across all predictors in the account. Forecasts are billed to the nearest 1,000 and charged on a per 1,000 basis.
TimeSeriesForecastsGenerated	PredictorArn ForecastArn	Count	Average, Sum, Min, Max	The number of unique time series forecasts generated for each quantile across all predictors in the account. Forecasts are billed to the nearest 1,000 and charged on a per 1,000 basis.

Compliance Validation for Amazon Forecast

Third-party auditors assess the security and compliance of Amazon Forecast as part of multiple AWS compliance programs. These include SOC, PCI, HIPAA, and others.

For a list of AWS services in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using Forecast is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- Architecting for HIPAA Security and Compliance Whitepaper – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.

- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

Resilience in Amazon Forecast

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure Security in Amazon Forecast

As a managed service, Amazon Forecast is protected by the AWS global network security procedures that are described in the Amazon Web Services: Overview of Security Processes whitepaper.

You use AWS published API calls to access Forecast through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

Guidelines and Limits

The following sections contain information about Amazon Forecast guidelines and limits.

Topics
- Supported AWS Regions (p. 89)
- Compliance (p. 89)
- Service Limits (p. 89)

Supported AWS Regions

For a list of AWS Regions that support Forecast, see AWS Regions and Endpoints in the *Amazon Web Services General Reference*.

Compliance

For more information about Forecast compliance programs, see AWS Compliance, AWS Compliance Programs, and AWS Services in Scope by Compliance Program.

Service Limits

Forecast has the following service limits.

Limits Imposed by the CreateDatasetImportJob (p. 120) API

Resource	Default Limit
Maximum number of files in your Amazon S3 bucket	10,000
Maximum cumulative size of all files in your Amazon S3 bucket	5 GB
Maximum number of datasets in a dataset group	3 (1 for each type)
Maximum number of rows in a dataset	100 million
Maximum number of columns in a target time series dataset (required columns + additional forecast dimensions)	13 (3 + 10)
Maximum number of columns in a related time series dataset (required columns + additional forecast dimensions + related features)	25 (2 + 10 + 13)

Resource	Default Limit
Maximum number of columns in an item metadata dataset	10

Limits Imposed by the CreatePredictor (p. 129) API

Resource	Default Limit
Maximum `NumberOfBacktestWindows` (EvaluationParameters (p. 206))	5
Maximum number of forecasts per predictor (number of items X number of unique values across forecast dimensions in the target time series dataset)	100,000 across all target time series dimensions (For instance, you can have 100,000 items, or 10 items in 10,00 locations, or 10 items in 100 warehouses in 100 cities.)
Forecast horizon	The lesser of 500 data points or 1/3 of the target time series dataset length

General Resource Limits

Resource	Default Limit
Maximum parallel running `CreateDatasetImportJob` tasks	3
Maximum parallel running `CreatePredictor` tasks	3
Maximum parallel running CreateForecast (p. 123) tasks	3
Maximum number of dataset import jobs	1000
Maximum number of predictors	500
Maximum number of forecasts	10
Maximum number of forecast export jobs	1000
Maximum number of parallel forecast export jobs	3
Maximum time for which a forecast can be queried on (console or API)	30 days

Reserved Field Names

Amazon Forecast reserves the following names. You can't use these names for your schema fields or dataset headers.

A

- `A`
- `ABORT`
- `ABS`
- `ABSOLUTE`
- `ACCESS`
- `ACTION`
- `ADA`
- `ADD`
- `ADMIN`
- `AFTER`
- `AGGREGATE`
- `ALIAS`
- `ALL`
- `ALLOCATE`
- `ALSO`
- `ALTER`
- `ALWAYS`
- `ANALYSE`
- `ANALYZE`
- `AND`
- `ANY`
- `ARE`
- `ARRAY`
- `AS`
- `ASC`
- `ASENSITIVE`
- `ASSERTION`
- `ASSIGNMENT`
- `ASYMMETRIC`
- `AT`
- `ATOMIC`
- `ATTRIBUTE`
- `ATTRIBUTES`
- `AUDIT`
- `AUTHORIZATION`
- `AUTO_INCREMENT`
- `AVG`

- AVG_ROW_LENGTH

B

- BACKUP
- BACKWARD
- BEFORE
- BEGIN
- BERNOULLI
- BETWEEN
- BIGINT
- BINARY
- BIT
- BIT_LENGTH
- BITVAR
- BLOB
- BOOL
- BOOLEAN
- BOTH
- BREADTH
- BREAK
- BROWSE
- BULK
- BY

C

- C
- CACHE
- CALL
- CALLED
- CARDINALITY
- CASCADE
- CASCADED
- CASE
- CAST
- CATALOG
- CATALOG_NAME
- CEIL
- CEILING
- CHAIN
- CHANGE
- CHAR
- CHAR_LENGTH
- CHARACTER
- CHARACTER_LENGTH

- `CHARACTER_SET_CATALOG`
- `CHARACTER_SET_NAME`
- `CHARACTER_SET_SCHEMA`
- `CHARACTERISTICS`
- `CHARACTERS`
- `CHECK`
- `CHECKED`
- `CHECKPOINT`
- `CHECKSUM`
- `CLASS`
- `CLASS_ORIGIN`
- `CLOB`
- `CLOSE`
- `CLUSTER`
- `CLUSTERED`
- `COALESCE`
- `COBOL`
- `COLLATE`
- `COLLATION`
- `COLLATION_CATALOG`
- `COLLATION_NAME`
- `COLLATION_SCHEMA`
- `COLLECT`
- `COLUMN`
- `COLUMN_NAME`
- `COLUMNS`
- `COMMAND_FUNCTION`
- `COMMAND_FUNCTION_CODE`
- `COMMENT`
- `COMMIT`
- `COMMITTED`
- `COMPLETION`
- `COMPRESS`
- `COMPUTE`
- `CONDITION`
- `CONDITION_NUMBER`
- `CONNECT`
- `CONNECTION`
- `CONNECTION_NAME`
- `CONSTRAINT`
- `CONSTRAINT_CATALOG`
- `CONSTRAINT_NAME`
- `CONSTRAINT_SCHEMA`
- `CONSTRAINTS`
- `CONSTRUCTOR`

- CONTAINS
- CONTAINSTABLE
- CONTINUE
- CONVERSION
- CONVERT
- COPY
- CORR
- CORRESPONDING
- COUNT
- COVAR_POP
- COVAR_SAMP
- CREATE
- CREATEDB
- CREATEROLE
- CREATEUSER
- CROSS
- CSV
- CUBE
- CUME_DIST
- CURRENT
- CURRENT_DATE
- CURRENT_DEFAULT_TRANSFORM_GROUP
- CURRENT_PATH
- CURRENT_ROLE
- CURRENT_TIME
- CURRENT_TIMESTAMP
- CURRENT_TRANSFORM_GROUP_FOR_TYPE
- CURRENT_USER
- CURSOR
- CURSOR_NAME
- CYCLE

D

- DATA
- DATABASE
- DATABASES
- DATETIME
- DATETIME_INTERVAL_CODE
- DATETIME_INTERVAL_PRECISION
- DAY
- DAY_HOUR
- DAY_MICROSECOND
- DAY_MINUTE
- DAY_SECOND
- DAYOFMONTH

- DAYOFWEEK
- DAYOFYEAR
- DBCC
- DEALLOCATE
- DEC
- DECIMAL
- DECLARE
- DEFAULT
- DEFAULTS
- DEFERRABLE
- DEFERRED
- DEFINED
- DEFINER
- DEGREE
- DELAY_KEY_WRITE
- DELAYED
- DELETE
- DELIMITER
- DELIMITERS
- DENSE_RANK
- DENY
- DEPTH
- DEREF
- DERIVED
- DESC
- DESCRIBE
- DESCRIPTOR
- DESTROY
- DESTRUCTOR
- DETERMINISTIC
- DIAGNOSTICS
- DICTIONARY
- DISABLE
- DISCONNECT
- DISK
- DISPATCH
- DISTINCT
- DISTINCTROW
- DISTRIBUTED
- DIV
- DO
- DOMAIN
- DOUBLE
- DROP
- DUAL
- DUMMY

- DUMP
- DYNAMIC
- DYNAMIC_FUNCTION
- DYNAMIC_FUNCTION_CODE

E

- EACH
- ELEMENT
- ELSE
- ELSEIF
- ENABLE
- ENCLOSED
- ENCODING
- ENCRYPTED
- END
- END-EXEC
- ENUM
- EQUALS
- ERRLVL
- ESCAPE
- ESCAPED
- EVERY
- EXCEPT
- EXCEPTION
- EXCLUDE
- EXCLUDING
- EXCLUSIVE
- EXEC
- EXECUTE
- EXISTING
- EXISTS
- EXIT
- EXP
- EXPLAIN
- EXTERNAL
- EXTRACT

F

- FALSE
- FETCH
- FIELDS
- FILE
- FILLFACTOR
- FILTER

- FINAL
- FIRST
- FLOAT
- FLOAT4
- FLOAT8
- FLOOR
- FLUSH
- FOLLOWING
- FOR
- FORCE
- FOREIGN
- FORTRAN
- FORWARD
- FOUND
- FREE
- FREETEXT
- FREETEXTTABLE
- FREEZE
- FROM
- FULL
- FULLTEXT
- FUNCTION
- FUSION

G

- G
- GENERAL
- GENERATED
- GET
- GLOBAL
- GO
- GOTO
- GRANT
- GRANTED
- GRANTS
- GREATEST
- GROUP
- GROUPING

H

- HANDLER
- HAVING
- HEADER
- HEAP

- `HIERARCHY`
- `HIGH_PRIORITY`
- `HOLD`
- `HOLDLOCK`
- `HOST`
- `HOSTS`
- `HOUR`
- `HOUR_MICROSECOND`
- `HOUR_MINUTE`
- `HOUR_SECOND`

I

- `IDENTIFIED`
- `IDENTITY`
- `IDENTITY_INSERT`
- `IDENTITYCOL`
- `IF`
- `IGNORE`
- `ILIKE`
- `IMMEDIATE`
- `IMMUTABLE`
- `IMPLEMENTATION`
- `IMPLICIT`
- `IN`
- `INCLUDE`
- `INCLUDING`
- `INCREMENT`
- `INDEX`
- `INDICATOR`
- `INFILE`
- `INFIX`
- `INHERIT`
- `INHERITS`
- `INITIAL`
- `INITIALIZE`
- `INITIALLY`
- `INNER`
- `INOUT`
- `INPUT`
- `INSENSITIVE`
- `INSERT`
- `INSERT_ID`
- `INSTANCE`
- `INSTANTIABLE`
- `INSTEAD`

- `INT`
- `INT1`
- `INT2`
- `INT3`
- `INT4`
- `INT8`
- `INTEGER`
- `INTERSECT`
- `INTERSECTION`
- `INTERVAL`
- `INTO`
- `INVOKER`
- `IS`
- `ISAM`
- `ISNULL`
- `ISOLATION`
- `ITERATE`

J

- `JOIN`

K

- `K`
- `KEY`
- `KEY_MEMBER`
- `KEY_TYPE`
- `KEYS`
- `KILL`

L

- `LANCOMPILER`
- `LANGUAGE`
- `LARGE`
- `LAST`
- `LAST_INSERT_ID`
- `LATERAL`
- `LEADING`
- `LEAST`
- `LEAVE`
- `LEFT`
- `LENGTH`
- `LESS`
- `LEVEL`
- `LIKE`

- `LIMIT`
- `LINENO`
- `LINES`
- `LISTEN`
- `LN`
- `LOAD`
- `LOCAL`
- `LOCALTIME`
- `LOCALTIMESTAMP`
- `LOCATOR`
- `LOCK`
- `LOGIN`
- `LOGS`
- `LONG`
- `LONGBLOB`
- `LONGTEXT`
- `LOOP`
- `LOW_PRIORITY`
- `LOWER`

M

- `M`
- `MAP`
- `MATCH`
- `MATCHED`
- `MAX`
- `MAX_ROWS`
- `MAXEXTENTS`
- `MAXVALUE`
- `MEDIUMBLOB`
- `MEDIUMINT`
- `MEDIUMTEXT`
- `MEMBER`
- `MERGE`
- `MESSAGE_LENGTH`
- `MESSAGE_OCTET_LENGTH`
- `MESSAGE_TEXT`
- `METHOD`
- `MIDDLEINT`
- `MIN`
- `MIN_ROWS`
- `MINUS`
- `MINUTE`
- `MINUTE_MICROSECOND`
- `MINUTE_SECOND`

- MINVALUE
- MLSLABEL
- MOD
- MODE
- MODIFIES
- MODIFY
- MODULE
- MONTH
- MONTHNAME
- MORE
- MOVE
- MULTISET
- MUMPS
- MYISAM

N

- NAME
- NAMES
- NATIONAL
- NATURAL
- NCHAR
- NCLOB
- NESTING
- NEW
- NEXT
- NO
- NO_WRITE_TO_BINLOG
- NOAUDIT
- NOCHECK
- NOCOMPRESS
- NOCREATEDB
- NOCREATEROLE
- NOCREATEUSER
- NOINHERIT
- NOLOGIN
- NONCLUSTERED
- NONE
- NORMALIZE
- NORMALIZED
- NOSUPERUSER
- NOT
- NOTHING
- NOTIFY
- NOTNULL
- NOWAIT

- NULL
- NULLABLE
- NULLIF
- NULLS
- NUMBER
- NUMERIC

O

- OBJECT
- OCTET_LENGTH
- OCTETS
- OF
- OFF
- OFFLINE
- OFFSET
- OFFSETS
- OIDS
- OLD
- ON
- ONLINE
- ONLY
- OPEN
- OPENDATASOURCE
- OPENQUERY
- OPENROWSET
- OPENXML
- OPERATION
- OPERATOR
- OPTIMIZE
- OPTION
- OPTIONALLY
- OPTIONS
- OR
- ORDER
- ORDERING
- ORDINALITY
- OTHERS
- OUT
- OUTER
- OUTFILE
- OUTPUT
- OVER
- OVERLAPS
- OVERLAY
- OVERRIDING

- OWNER

P

- PACK_KEYS
- PAD
- PARAMETER
- PARAMETER_MODE
- PARAMETER_NAME
- PARAMETER_ORDINAL_POSITION
- PARAMETER_SPECIFIC_CATALOG
- PARAMETER_SPECIFIC_NAME
- PARAMETER_SPECIFIC_SCHEMA
- PARAMETERS
- PARTIAL
- PARTITION
- PASCAL
- PASSWORD
- PATH
- PCTFREE
- PERCENT
- PERCENT_RANK
- PERCENTILE_CONT
- PERCENTILE_DISC
- PLACING
- PLAN
- PLI
- POSITION
- POSTFIX
- POWER
- PRECEDING
- PRECISION
- PREFIX
- PREORDER
- PREPARE
- PREPARED
- PRESERVE
- PRIMARY
- PRINT
- PRIOR
- PRIVILEGES
- PROC
- PROCEDURAL
- PROCEDURE
- PROCESS
- PROCESSLIST

- PUBLIC
- PURGE

Q

- QUOTE

R

- RAID0
- RAISERROR
- RANGE
- RANK
- RAW
- READ
- READS
- READTEXT
- REAL
- RECHECK
- RECONFIGURE
- RECURSIVE
- REF
- REFERENCES
- REFERENCING
- REGEXP
- REGR_AVGX
- REGR_AVGY
- REGR_COUNT
- REGR_INTERCEPT
- REGR_R2
- REGR_SLOPE
- REGR_SXX
- REGR_SXY
- REGR_SYY
- REINDEX
- RELATIVE
- RELEASE
- RELOAD
- RENAME
- REPEAT
- REPEATABLE
- REPLACE
- REPLICATION
- REQUIRE
- RESET
- RESIGNAL

- RESOURCE
- RESTART
- RESTORE
- RESTRICT
- RESULT
- RETURN
- RETURNED_CARDINALITY
- RETURNED_LENGTH
- RETURNED_OCTET_LENGTH
- RETURNED_SQLSTATE
- RETURNS
- REVOKE
- RIGHT
- RLIKE
- ROLE
- ROLLBACK
- ROLLUP
- ROUTINE
- ROUTINE_CATALOG
- ROUTINE_NAME
- ROUTINE_SCHEMA
- ROW
- ROW_COUNT
- ROW_NUMBER
- ROWCOUNT
- ROWGUIDCOL
- ROWID
- ROWNUM
- ROWS
- RULE

S

- SAVE
- SAVEPOINT
- SCALE
- SCHEMA
- SCHEMA_NAME
- SCHEMAS
- SCOPE
- SCOPE_CATALOG
- SCOPE_NAME
- SCOPE_SCHEMA
- SCROLL
- SEARCH
- SECOND

- SECOND_MICROSECOND
- SECTION
- SECURITY
- SELECT
- SELF
- SENSITIVE
- SEPARATOR
- SEQUENCE
- SERIALIZABLE
- SERVER_NAME
- SESSION
- SESSION_USER
- SET
- SETOF
- SETS
- SETUSER
- SHARE
- SHOW
- SHUTDOWN
- SIGNAL
- SIMILAR
- SIMPLE
- SIZE
- SMALLINT
- SOME
- SONAME
- SOURCE
- SPACE
- SPATIAL
- SPECIFIC
- SPECIFIC_NAME
- SPECIFICTYPE
- SQL
- SQL_BIG_RESULT
- SQL_BIG_SELECTS
- SQL_BIG_TABLES
- SQL_CALC_FOUND_ROWS
- SQL_LOG_OFF
- SQL_LOG_UPDATE
- SQL_LOW_PRIORITY_UPDATES
- SQL_SELECT_LIMIT
- SQL_SMALL_RESULT
- SQL_WARNINGS
- SQLCA
- SQLCODE
- SQLERROR

- SQLEXCEPTION
- SQLSTATE
- SQLWARNING
- SQRT
- SSL
- STABLE
- START
- STARTING
- STATE
- STATEMENT
- STATIC
- STATISTICS
- STATUS
- STDDEV_POP
- STDDEV_SAMP
- STDIN
- STDOUT
- STORAGE
- STRAIGHT_JOIN
- STRICT
- STRING
- STRUCTURE
- STYLE
- SUBCLASS_ORIGIN
- SUBLIST
- SUBMULTISET
- SUBSTRING
- SUCCESSFUL
- SUM
- SUPERUSER
- SYMMETRIC
- SYNONYM
- SYSDATE
- SYSID
- SYSTEM
- SYSTEM_USER

T

- TABLE
- TABLE_NAME
- TABLES
- TABLESAMPLE
- TABLESPACE
- TEMP
- TEMPLATE

- `TEMPORARY`
- `TERMINATE`
- `TERMINATED`
- `TEXT`
- `TEXTSIZE`
- `THAN`
- `THEN`
- `TIES`
- `TIME`
- `TIMEZONE_HOUR`
- `TIMEZONE_MINUTE`
- `TINYBLOB`
- `TINYINT`
- `TINYTEXT`
- `TO`
- `TOAST`
- `TOP`
- `TOP_LEVEL_COUNT`
- `TRAILING`
- `TRAN`
- `TRANSACTION`
- `TRANSACTION_ACTIVE`
- `TRANSACTIONS_COMMITTED`
- `TRANSACTIONS_ROLLED_BACK`
- `TRANSFORM`
- `TRANSFORMS`
- `TRANSLATE`
- `TRANSLATION`
- `TREAT`
- `TRIGGER`
- `TRIGGER_CATALOG`
- `TRIGGER_NAME`
- `TRIGGER_SCHEMA`
- `TRIM`
- `TRUE`
- `TRUNCATE`
- `TRUSTED`
- `TSEQUAL`
- `TYPE`

U

- `UESCAPE`
- `UID`
- `UNBOUNDED`
- `UNCOMMITTED`

- UNDER
- UNDO
- UNENCRYPTED
- UNION
- UNIQUE
- UNKNOWN
- UNLISTEN
- UNLOCK
- UNNAMED
- UNNEST
- UNSIGNED
- UNTIL
- UPDATE
- UPDATETEXT
- UPPER
- USAGE
- USE
- USER
- USER_DEFINED_TYPE_CATALOG
- USER_DEFINED_TYPE_CODE
- USER_DEFINED_TYPE_NAME
- USER_DEFINED_TYPE_SCHEMA
- USING
- UTC_DATE
- UTC_TIME
- UTC_TIMESTAMP

V

- VACUUM
- VALID
- VALIDATE
- VALIDATOR
- VALUE
- VALUES
- VAR_POP
- VAR_SAMP
- VARBINARY
- VARCHAR
- VARCHAR2
- VARCHARACTER
- VARIABLE
- VARIABLES
- VARYING
- VERBOSE
- VIEW

- VOLATILE

W

- WAITFOR
- WHEN
- WHENEVER
- WHERE
- WHILE
- WIDTH_BUCKET
- WINDOW
- WITH
- WITHIN
- WITHOUT
- WORK
- WRITE
- WRITETEXT

X

- X509
- XOR

Y

- YEAR
- YEAR_MONTH

Z

- ZEROFILL
- ZONE

API Reference

This section provides documentation for the Amazon Forecast API operations.

Topics

- Actions (p. 111)
- Data Types (p. 192)
- Common Errors (p. 241)
- Common Parameters (p. 243)

Actions

The following actions are supported by Amazon Forecast Service:

- CreateDataset (p. 113)
- CreateDatasetGroup (p. 117)
- CreateDatasetImportJob (p. 120)
- CreateForecast (p. 123)
- CreateForecastExportJob (p. 126)
- CreatePredictor (p. 129)
- DeleteDataset (p. 135)
- DeleteDatasetGroup (p. 137)
- DeleteDatasetImportJob (p. 139)
- DeleteForecast (p. 141)
- DeleteForecastExportJob (p. 143)
- DeletePredictor (p. 145)
- DescribeDataset (p. 147)
- DescribeDatasetGroup (p. 151)
- DescribeDatasetImportJob (p. 154)
- DescribeForecast (p. 158)
- DescribeForecastExportJob (p. 161)
- DescribePredictor (p. 164)
- GetAccuracyMetrics (p. 170)
- ListDatasetGroups (p. 172)
- ListDatasetImportJobs (p. 174)
- ListDatasets (p. 177)
- ListForecastExportJobs (p. 179)
- ListForecasts (p. 182)
- ListPredictors (p. 185)
- UpdateDatasetGroup (p. 188)

The following actions are supported by Amazon Forecast Query Service:

- QueryForecast (p. 190)

Amazon Forecast Service

The following actions are supported by Amazon Forecast Service:

- CreateDataset (p. 113)
- CreateDatasetGroup (p. 117)
- CreateDatasetImportJob (p. 120)
- CreateForecast (p. 123)
- CreateForecastExportJob (p. 126)
- CreatePredictor (p. 129)
- DeleteDataset (p. 135)
- DeleteDatasetGroup (p. 137)
- DeleteDatasetImportJob (p. 139)
- DeleteForecast (p. 141)
- DeleteForecastExportJob (p. 143)
- DeletePredictor (p. 145)
- DescribeDataset (p. 147)
- DescribeDatasetGroup (p. 151)
- DescribeDatasetImportJob (p. 154)
- DescribeForecast (p. 158)
- DescribeForecastExportJob (p. 161)
- DescribePredictor (p. 164)
- GetAccuracyMetrics (p. 170)
- ListDatasetGroups (p. 172)
- ListDatasetImportJobs (p. 174)
- ListDatasets (p. 177)
- ListForecastExportJobs (p. 179)
- ListForecasts (p. 182)
- ListPredictors (p. 185)
- UpdateDatasetGroup (p. 188)

CreateDataset

Service: Amazon Forecast Service

Creates an Amazon Forecast dataset. The information about the dataset that you provide helps Forecast understand how to consume the data for model training. This includes the following:

- `DataFrequency` - How frequently your historical time-series data is collected.
- `Domain` and `DatasetType` - Each dataset has an associated dataset domain and a type within the domain. Amazon Forecast provides a list of predefined domains and types within each domain. For each unique dataset domain and type within the domain, Amazon Forecast requires your data to include a minimum set of predefined fields.
- `Schema` - A schema specifies the fields in the dataset, including the field name and data type.

After creating a dataset, you import your training data into it and add the dataset to a dataset group. You use the dataset group to create a predictor. For more information, see Datasets and Dataset Groups (p. 2).

To get a list of all your datasets, use the ListDatasets (p. 177) operation.

For example Forecast datasets, see the Amazon Forecast Sample GitHub repository.

> **Note**
> The `Status` of a dataset must be `ACTIVE` before you can import training data. Use the DescribeDataset (p. 147) operation to get the status.

Request Syntax

```
{
   "DataFrequency": "string",
   "DatasetName": "string",
   "DatasetType": "string",
   "Domain": "string",
   "EncryptionConfig": {
      "KMSKeyArn": "string",
      "RoleArn": "string"
   },
   "Schema": {
      "Attributes": [
         {
            "AttributeName": "string",
            "AttributeType": "string"
         }
      ]
   }
}
```

Request Parameters

The request accepts the following data in JSON format.

DataFrequency (p. 113)

The frequency of data collection. This parameter is required for RELATED_TIME_SERIES datasets.

Valid intervals are Y (Year), M (Month), W (Week), D (Day), H (Hour), 30min (30 minutes), 15min (15 minutes), 10min (10 minutes), 5min (5 minutes), and 1min (1 minute). For example, "D" indicates every day and "15min" indicates every 15 minutes.

Type: String

Pattern: `^Y|M|W|D|H|30min|15min|10min|5min|1min$`

Required: No

DatasetName (p. 113)

A name for the dataset.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

DatasetType (p. 113)

The dataset type. Valid values depend on the chosen `Domain`.

Type: String

Valid Values: `TARGET_TIME_SERIES | RELATED_TIME_SERIES | ITEM_METADATA`

Required: Yes

Domain (p. 113)

The domain associated with the dataset. When you add a dataset to a dataset group, this value and the value specified for the `Domain` parameter of the CreateDatasetGroup (p. 117) operation must match.

The `Domain` and `DatasetType` that you choose determine the fields that must be present in the training data that you import to the dataset. For example, if you choose the `RETAIL` domain and `TARGET_TIME_SERIES` as the `DatasetType`, Amazon Forecast requires `item_id`, `timestamp`, and `demand` fields to be present in your data. For more information, see Datasets and Dataset Groups (p. 2).

Type: String

Valid Values: `RETAIL | CUSTOM | INVENTORY_PLANNING | EC2_CAPACITY | WORK_FORCE | WEB_TRAFFIC | METRICS`

Required: Yes

EncryptionConfig (p. 113)

An AWS Key Management Service (KMS) key and the AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key.

Type: EncryptionConfig (p. 205) object

Required: No

Schema (p. 113)

The schema for the dataset. The schema attributes and their order must match the fields in your data. The dataset `Domain` and `DatasetType` that you choose determine the minimum required fields in your training data. For information about the required fields for a specific dataset domain and type, see Predefined Dataset Domains and Dataset Types (p. 46).

Type: Schema (p. 230) object

Required: Yes

Response Syntax

```
{
    "DatasetArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DatasetArn (p. 115)

The Amazon Resource Name (ARN) of the dataset.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateDatasetGroup

Service: Amazon Forecast Service

Creates a dataset group, which holds a collection of related datasets. You can add datasets to the dataset group when you create the dataset group, or later by using the UpdateDatasetGroup (p. 188) operation.

After creating a dataset group and adding datasets, you use the dataset group when you create a predictor. For more information, see Datasets and Dataset Groups (p. 2).

To get a list of all your datasets groups, use the ListDatasetGroups (p. 172) operation.

> **Note**
> The `Status` of a dataset group must be `ACTIVE` before you can create use the dataset group to create a predictor. To get the status, use the DescribeDatasetGroup (p. 151) operation.

Request Syntax

```
{
   "DatasetArns": [ "string" ],
   "DatasetGroupName": "string",
   "Domain": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetArns (p. 117)

An array of Amazon Resource Names (ARNs) of the datasets that you want to include in the dataset group.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

DatasetGroupName (p. 117)

A name for the dataset group.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Domain (p. 117)

The domain associated with the dataset group. When you add a dataset to a dataset group, this value and the value specified for the `Domain` parameter of the CreateDataset (p. 113) operation must match.

The `Domain` and `DatasetType` that you choose determine the fields that must be present in training data that you import to a dataset. For example, if you choose the `RETAIL` domain and `TARGET_TIME_SERIES` as the `DatasetType`, Amazon Forecast requires that `item_id`, `timestamp`,

and `demand` fields are present in your data. For more information, see Datasets and Dataset Groups (p. 2).

Type: String

Valid Values: `RETAIL | CUSTOM | INVENTORY_PLANNING | EC2_CAPACITY | WORK_FORCE | WEB_TRAFFIC | METRICS`

Required: Yes

Response Syntax

```
{
    "DatasetGroupArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DatasetGroupArn (p. 118)

The Amazon Resource Name (ARN) of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateDatasetImportJob

Service: Amazon Forecast Service

Imports your training data to an Amazon Forecast dataset. You provide the location of your training data in an Amazon Simple Storage Service (Amazon S3) bucket and the Amazon Resource Name (ARN) of the dataset that you want to import the data to.

You must specify a DataSource (p. 204) object that includes an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the data. For more information, see Set Up Permissions for Amazon Forecast (p. 15).

The training data must be in CSV format. The delimiter must be a comma (,).

You can specify the path to a specific CSV file, the S3 bucket, or to a folder in the S3 bucket. For the latter two cases, Amazon Forecast imports all files up to the limit of 10,000 files.

Because dataset imports are not aggregated, your most recent dataset import is the one that is used when training a predictor or generating a forecast. Make sure that your most recent dataset import contains all of the data you want to model off of, and not just the new data collected since the previous import.

To get a list of all your dataset import jobs, filtered by specified criteria, use the ListDatasetImportJobs (p. 174) operation.

Request Syntax

```
{
    "DatasetArn": "string",
    "DatasetImportJobName": "string",
    "DataSource": {
        "S3Config": {
            "KMSKeyArn": "string",
            "Path": "string",
            "RoleArn": "string"
        }
    },
    "TimestampFormat": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetArn (p. 120)

The Amazon Resource Name (ARN) of the Amazon Forecast dataset that you want to import data to.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

DatasetImportJobName (p. 120)

The name for the dataset import job. We recommend including the current timestamp in the name, for example, `20190721DatasetImport`. This can help you avoid getting a `ResourceAlreadyExistsException` exception.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

DataSource (p. 120)

The location of the training data to import and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the data. The training data must be stored in an Amazon S3 bucket.

If encryption is used, `DataSource` must include an AWS Key Management Service (KMS) key and the IAM role must allow Amazon Forecast permission to access the key. The KMS key and IAM role must match those specified in the `EncryptionConfig` parameter of the CreateDataset (p. 113) operation.

Type: DataSource (p. 204) object

Required: Yes

TimestampFormat (p. 120)

The format of timestamps in the dataset. The format that you specify depends on the `DataFrequency` specified when the dataset was created. The following formats are supported

- "yyyy-MM-dd"

  For the following data frequencies: Y, M, W, and D

- "yyyy-MM-dd HH:mm:ss"

  For the following data frequencies: H, 30min, 15min, and 1min; and optionally, for: Y, M, W, and D

If the format isn't specified, Amazon Forecast expects the format to be "yyyy-MM-dd HH:mm:ss".

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\:\.\,\'\s]+$`

Required: No

Response Syntax

```
{
   "DatasetImportJobArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DatasetImportJobArn (p. 121)

The Amazon Resource Name (ARN) of the dataset import job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateForecast

Service: Amazon Forecast Service

Creates a forecast for each item in the `TARGET_TIME_SERIES` dataset that was used to train the predictor. This is known as inference. To retrieve the forecast for a single item at low latency, use the QueryForecast (p. 190) operation. To export the complete forecast into your Amazon Simple Storage Service (Amazon S3) bucket, use the CreateForecastExportJob (p. 126) operation.

The range of the forecast is determined by the `ForecastHorizon` value, which you specify in the CreatePredictor (p. 129) request. When you query a forecast, you can request a specific date range within the forecast.

To get a list of all your forecasts, use the ListForecasts (p. 182) operation.

> **Note**
> The forecasts generated by Amazon Forecast are in the same time zone as the dataset that was used to create the predictor.

For more information, see Forecasts (p. 13).

> **Note**
> The `Status` of the forecast must be `ACTIVE` before you can query or export the forecast. Use the DescribeForecast (p. 158) operation to get the status.

Request Syntax

```
{
   "ForecastName": "string",
   "ForecastTypes": [ "string" ],
   "PredictorArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastName (p. 123)

A name for the forecast.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

ForecastTypes (p. 123)

The quantiles at which probabilistic forecasts are generated. **You can currently specify up to 5 quantiles per forecast**. Accepted values include `0.01 to 0.99` (increments of .01 only) and `mean`. The mean forecast is different from the median (0.50) when the distribution is not symmetric (for example, Beta and Negative Binomial). The default value is `["0.1", "0.5", "0.9"]`.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Pattern: `(^0?\.\d\d?$|^mean$)`

Required: No

PredictorArn (p. 123)

The Amazon Resource Name (ARN) of the predictor to use to generate the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

Response Syntax

```
{
   "ForecastArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ForecastArn (p. 124)

The Amazon Resource Name (ARN) of the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreateForecastExportJob

Service: Amazon Forecast Service

Exports a forecast created by the CreateForecast (p. 123) operation to your Amazon Simple Storage Service (Amazon S3) bucket. The forecast file name will match the following conventions:

<ForecastExportJobName>_<ExportTimestamp>_<PageNumber>

where the <ExportTimestamp> component is in Java SimpleDateFormat (yyyy-MM-ddTHH-mm-ssZ).

You must specify a DataDestination (p. 198) object that includes an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the Amazon S3 bucket. For more information, see Set Up Permissions for Amazon Forecast (p. 15).

For more information, see Forecasts (p. 13).

To get a list of all your forecast export jobs, use the ListForecastExportJobs (p. 179) operation.

> **Note**
> The `Status` of the forecast export job must be `ACTIVE` before you can access the forecast in your Amazon S3 bucket. To get the status, use the DescribeForecastExportJob (p. 161) operation.

Request Syntax

```
{
    "Destination": {
        "S3Config": {
            "KMSKeyArn": "string",
            "Path": "string",
            "RoleArn": "string"
        }
    },
    "ForecastArn": "string",
    "ForecastExportJobName": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Destination (p. 126)

The location where you want to save the forecast and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the location. The forecast must be exported to an Amazon S3 bucket.

If encryption is used, `Destination` must include an AWS Key Management Service (KMS) key. The IAM role must allow Amazon Forecast permission to access the key.

Type: DataDestination (p. 198) object

Required: Yes

ForecastArn (p. 126)

The Amazon Resource Name (ARN) of the forecast that you want to export.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

ForecastExportJobName (p. 126)

The name for the forecast export job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Response Syntax

```
{
    "ForecastExportJobArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ForecastExportJobArn (p. 127)

The Amazon Resource Name (ARN) of the export job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

CreatePredictor

Service: Amazon Forecast Service

Creates an Amazon Forecast predictor.

In the request, you provide a dataset group and either specify an algorithm or let Amazon Forecast choose the algorithm for you using AutoML. If you specify an algorithm, you also can override algorithm-specific hyperparameters.

Amazon Forecast uses the chosen algorithm to train a model using the latest version of the datasets in the specified dataset group. The result is called a predictor. You then generate a forecast using the CreateForecast (p. 123) operation.

After training a model, the `CreatePredictor` operation also evaluates it. To see the evaluation metrics, use the GetAccuracyMetrics (p. 170) operation. Always review the evaluation metrics before deciding to use the predictor to generate a forecast.

Optionally, you can specify a featurization configuration to fill and aggregate the data fields in the `TARGET_TIME_SERIES` dataset to improve model training. For more information, see FeaturizationConfig (p. 209).

For RELATED_TIME_SERIES datasets, `CreatePredictor` verifies that the `DataFrequency` specified when the dataset was created matches the `ForecastFrequency`. TARGET_TIME_SERIES datasets don't have this restriction. Amazon Forecast also verifies the delimiter and timestamp format. For more information, see Datasets and Dataset Groups (p. 2).

AutoML

If you want Amazon Forecast to evaluate each algorithm and choose the one that minimizes the `objective function`, set `PerformAutoML` to `true`. The `objective function` is defined as the mean of the weighted p10, p50, and p90 quantile losses. For more information, see EvaluationResult (p. 207).

When AutoML is enabled, the following properties are disallowed:

- `AlgorithmArn`
- `HPOConfig`
- `PerformHPO`
- `TrainingParameters`

To get a list of all of your predictors, use the ListPredictors (p. 185) operation.

> **Note**
> Before you can use the predictor to create a forecast, the `Status` of the predictor must be `ACTIVE`, signifying that training has completed. To get the status, use the DescribePredictor (p. 164) operation.

Request Syntax

```
{
    "AlgorithmArn": "string",
    "EncryptionConfig": {
        "KMSKeyArn": "string",
        "RoleArn": "string"
    },
    "EvaluationParameters": {
        "BackTestWindowOffset": number,
        "NumberOfBacktestWindows": number
```

```
        },
        "FeaturizationConfig": {
            "Featurizations": [
                {
                    "AttributeName": "string",
                    "FeaturizationPipeline": [
                        {
                            "FeaturizationMethodName": "string",
                            "FeaturizationMethodParameters": {
                                "string" : "string"
                            }
                        }
                    ]
                }
            ],
            "ForecastDimensions": [ "string" ],
            "ForecastFrequency": "string"
        },
        "ForecastHorizon": number,
        "HPOConfig": {
            "ParameterRanges": {
                "CategoricalParameterRanges": [
                    {
                        "Name": "string",
                        "Values": [ "string" ]
                    }
                ],
                "ContinuousParameterRanges": [
                    {
                        "MaxValue": number,
                        "MinValue": number,
                        "Name": "string",
                        "ScalingType": "string"
                    }
                ],
                "IntegerParameterRanges": [
                    {
                        "MaxValue": number,
                        "MinValue": number,
                        "Name": "string",
                        "ScalingType": "string"
                    }
                ]
            }
        },
        "InputDataConfig": {
            "DatasetGroupArn": "string",
            "SupplementaryFeatures": [
                {
                    "Name": "string",
                    "Value": "string"
                }
            ]
        },
        "PerformAutoML": boolean,
        "PerformHPO": boolean,
        "PredictorName": "string",
        "TrainingParameters": {
            "string" : "string"
        }
}
```

Request Parameters

The request accepts the following data in JSON format.

AlgorithmArn (p. 129)

The Amazon Resource Name (ARN) of the algorithm to use for model training. Required if `PerformAutoML` is not set to `true`.

Supported algorithms:

- `arn:aws:forecast:::algorithm/ARIMA`
- `arn:aws:forecast:::algorithm/Deep_AR_Plus`

    Supports hyperparameter optimization (HPO)
- `arn:aws:forecast:::algorithm/ETS`
- `arn:aws:forecast:::algorithm/NPTS`
- `arn:aws:forecast:::algorithm/Prophet`

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

EncryptionConfig (p. 129)

An AWS Key Management Service (KMS) key and the AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key.

Type: EncryptionConfig (p. 205) object

Required: No

EvaluationParameters (p. 129)

Used to override the default evaluation parameters of the specified algorithm. Amazon Forecast evaluates a predictor by splitting a dataset into training data and testing data. The evaluation parameters define how to perform the split and the number of iterations.

Type: EvaluationParameters (p. 206) object

Required: No

FeaturizationConfig (p. 129)

The featurization configuration.

Type: FeaturizationConfig (p. 209) object

Required: Yes

ForecastHorizon (p. 129)

Specifies the number of time-steps that the model is trained to predict. The forecast horizon is also called the prediction length.

For example, if you configure a dataset for daily data collection (using the `DataFrequency` parameter of the CreateDataset (p. 113) operation) and set the forecast horizon to 10, the model returns predictions for 10 days.

The maximum forecast horizon is the lesser of 500 time-steps or 1/3 of the TARGET_TIME_SERIES dataset length.

Type: Integer

Required: Yes

HPOConfig (p. 129)

Provides hyperparameter override values for the algorithm. If you don't provide this parameter, Amazon Forecast uses default values. The individual algorithms specify which hyperparameters support hyperparameter optimization (HPO). For more information, see Choosing an Amazon Forecast Algorithm (p. 55).

If you included the `HPOConfig` object, you must set `PerformHPO` to true.

Type: HyperParameterTuningJobConfig (p. 218) object

Required: No

InputDataConfig (p. 129)

Describes the dataset group that contains the data to use to train the predictor.

Type: InputDataConfig (p. 219) object

Required: Yes

PerformAutoML (p. 129)

Whether to perform AutoML. When Amazon Forecast performs AutoML, it evaluates the algorithms it provides and chooses the best algorithm and configuration for your training dataset.

The default value is `false`. In this case, you are required to specify an algorithm.

Set `PerformAutoML` to `true` to have Amazon Forecast perform AutoML. This is a good option if you aren't sure which algorithm is suitable for your training data. In this case, `PerformHPO` must be false.

Type: Boolean

Required: No

PerformHPO (p. 129)

Whether to perform hyperparameter optimization (HPO). HPO finds optimal hyperparameter values for your training data. The process of performing HPO is known as running a hyperparameter tuning job.

The default value is `false`. In this case, Amazon Forecast uses default hyperparameter values from the chosen algorithm.

To override the default values, set `PerformHPO` to `true` and, optionally, supply the HyperParameterTuningJobConfig (p. 218) object. The tuning job specifies a metric to optimize, which hyperparameters participate in tuning, and the valid range for each tunable hyperparameter. In this case, you are required to specify an algorithm and `PerformAutoML` must be false.

The following algorithm supports HPO:

- DeepAR+

Type: Boolean

Required: No

PredictorName (p. 129)

A name for the predictor.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

TrainingParameters (p. 129)

The hyperparameters to override for model training. The hyperparameters that you can override are listed in the individual algorithms. For the list of supported algorithms, see Choosing an Amazon Forecast Algorithm (p. 55).

Type: String to string map

Key Length Constraints: Maximum length of 256.

Key Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]\,\\]+$`

Value Length Constraints: Maximum length of 256.

Value Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]\,\"\\\s]+$`

Required: No

Response Syntax

```
{
    "PredictorArn": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

PredictorArn (p. 133)

The Amazon Resource Name (ARN) of the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

LimitExceededException

The limit on the number of resources per account has been exceeded.

HTTP Status Code: 400

ResourceAlreadyExistsException

There is already a resource with this name. Try again with a different name.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteDataset

Service: Amazon Forecast Service

Deletes an Amazon Forecast dataset that was created using the CreateDataset (p. 113) operation. You can only delete datasets that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status use the DescribeDataset (p. 147) operation.

> **Note**
> Forecast does not automatically update any dataset groups that contain the deleted dataset. In order to update the dataset group, use the UpdateDatasetGroup (p. 188) operation, omitting the deleted dataset's ARN.

Request Syntax

```
{
    "DatasetArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetArn (p. 135)

The Amazon Resource Name (ARN) of the dataset to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteDatasetGroup

Service: Amazon Forecast Service

Deletes a dataset group created using the CreateDatasetGroup (p. 117) operation. You can only delete dataset groups that have a status of `ACTIVE`, `CREATE_FAILED`, or `UPDATE_FAILED`. To get the status, use the DescribeDatasetGroup (p. 151) operation.

This operation deletes only the dataset group, not the datasets in the group.

Request Syntax

```
{
    "DatasetGroupArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetGroupArn (p. 137)

The Amazon Resource Name (ARN) of the dataset group to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteDatasetImportJob

Service: Amazon Forecast Service

Deletes a dataset import job created using the CreateDatasetImportJob (p. 120) operation. You can delete only dataset import jobs that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribeDatasetImportJob (p. 154) operation.

Request Syntax

```
{
    "DatasetImportJobArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetImportJobArn (p. 139)

The Amazon Resource Name (ARN) of the dataset import job to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteForecast

Service: Amazon Forecast Service

Deletes a forecast created using the CreateForecast (p. 123) operation. You can delete only forecasts that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribeForecast (p. 158) operation.

You can't delete a forecast while it is being exported. After a forecast is deleted, you can no longer query the forecast.

Request Syntax

```
{
    "ForecastArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastArn (p. 141)

The Amazon Resource Name (ARN) of the forecast to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeleteForecastExportJob

Service: Amazon Forecast Service

Deletes a forecast export job created using the CreateForecastExportJob (p. 126) operation. You can delete only export jobs that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribeForecastExportJob (p. 161) operation.

Request Syntax

```
{
    "ForecastExportJobArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastExportJobArn (p. 143)

The Amazon Resource Name (ARN) of the forecast export job to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DeletePredictor

Service: Amazon Forecast Service

Deletes a predictor created using the CreatePredictor (p. 129) operation. You can delete only predictor that have a status of `ACTIVE` or `CREATE_FAILED`. To get the status, use the DescribePredictor (p. 164) operation.

Request Syntax

```
{
    "PredictorArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

PredictorArn (p. 145)

The Amazon Resource Name (ARN) of the predictor to delete.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeDataset

Service: Amazon Forecast Service

Describes an Amazon Forecast dataset created using the CreateDataset (p. 113) operation.

In addition to listing the parameters specified in the `CreateDataset` request, this operation includes the following dataset properties:

- `CreationTime`
- `LastModificationTime`
- `Status`

Request Syntax

```
{
    "DatasetArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetArn (p. 147)

The Amazon Resource Name (ARN) of the dataset.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

Response Syntax

```
{
    "CreationTime": number,
    "DataFrequency": "string",
    "DatasetArn": "string",
    "DatasetName": "string",
    "DatasetType": "string",
    "Domain": "string",
    "EncryptionConfig": {
        "KMSKeyArn": "string",
        "RoleArn": "string"
    },
    "LastModificationTime": number,
    "Schema": {
        "Attributes": [
            {
                "AttributeName": "string",
                "AttributeType": "string"
            }
        ]
    },
    "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime (p. 147)

>   When the dataset was created.

>   Type: Timestamp

DataFrequency (p. 147)

>   The frequency of data collection.

>   Valid intervals are Y (Year), M (Month), W (Week), D (Day), H (Hour), 30min (30 minutes), 15min (15 minutes), 10min (10 minutes), 5min (5 minutes), and 1min (1 minute). For example, "M" indicates every month and "30min" indicates every 30 minutes.

>   Type: String

>   Pattern: `^Y|M|W|D|H|30min|15min|10min|5min|1min$`

DatasetArn (p. 147)

>   The Amazon Resource Name (ARN) of the dataset.

>   Type: String

>   Length Constraints: Maximum length of 256.

>   Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

DatasetName (p. 147)

>   The name of the dataset.

>   Type: String

>   Length Constraints: Minimum length of 1. Maximum length of 63.

>   Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

DatasetType (p. 147)

>   The dataset type.

>   Type: String

>   Valid Values: `TARGET_TIME_SERIES | RELATED_TIME_SERIES | ITEM_METADATA`

Domain (p. 147)

>   The domain associated with the dataset.

>   Type: String

>   Valid Values: `RETAIL | CUSTOM | INVENTORY_PLANNING | EC2_CAPACITY | WORK_FORCE | WEB_TRAFFIC | METRICS`

EncryptionConfig (p. 147)

>   The AWS Key Management Service (KMS) key and the AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key.

Type: EncryptionConfig (p. 205) object

LastModificationTime (p. 147)

When you create a dataset, `LastModificationTime` is the same as `CreationTime`. While data is being imported to the dataset, `LastModificationTime` is the current time of the `DescribeDataset` call. After a CreateDatasetImportJob (p. 120) operation has finished, `LastModificationTime` is when the import job completed or failed.

Type: Timestamp

Schema (p. 147)

An array of `SchemaAttribute` objects that specify the dataset fields. Each `SchemaAttribute` specifies the name and data type of a field.

Type: Schema (p. 230) object

Status (p. 147)

The status of the dataset. States include:

- `ACTIVE`
- `CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED`
- `DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED`
- `UPDATE_PENDING, UPDATE_IN_PROGRESS, UPDATE_FAILED`

The `UPDATE` states apply while data is imported to the dataset from a call to the CreateDatasetImportJob (p. 120) operation and reflect the status of the dataset import job. For example, when the import job status is `CREATE_IN_PROGRESS`, the status of the dataset is `UPDATE_IN_PROGRESS`.

> **Note**
> The `Status` of the dataset must be `ACTIVE` before you can import training data.

Type: String

Length Constraints: Maximum length of 256.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++

- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeDatasetGroup

Service: Amazon Forecast Service

Describes a dataset group created using the CreateDatasetGroup (p. 117) operation.

In addition to listing the parameters provided in the `CreateDatasetGroup` request, this operation includes the following properties:

- `DatasetArns` - The datasets belonging to the group.
- `CreationTime`
- `LastModificationTime`
- `Status`

Request Syntax

```
{
    "DatasetGroupArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetGroupArn (p. 151)

> The Amazon Resource Name (ARN) of the dataset group.
>
> Type: String
>
> Length Constraints: Maximum length of 256.
>
> Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`
>
> Required: Yes

Response Syntax

```
{
    "CreationTime": number,
    "DatasetArns": [ "string" ],
    "DatasetGroupArn": "string",
    "DatasetGroupName": "string",
    "Domain": "string",
    "LastModificationTime": number,
    "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime (p. 151)

> When the dataset group was created.

Type: Timestamp

DatasetArns (p. 151)

An array of Amazon Resource Names (ARNs) of the datasets contained in the dataset group.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

DatasetGroupArn (p. 151)

The ARN of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

DatasetGroupName (p. 151)

The name of the dataset group.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Domain (p. 151)

The domain associated with the dataset group.

Type: String

Valid Values: `RETAIL | CUSTOM | INVENTORY_PLANNING | EC2_CAPACITY | WORK_FORCE | WEB_TRAFFIC | METRICS`

LastModificationTime (p. 151)

When the dataset group was created or last updated from a call to the UpdateDatasetGroup (p. 188) operation. While the dataset group is being updated, `LastModificationTime` is the current time of the `DescribeDatasetGroup` call.

Type: Timestamp

Status (p. 151)

The status of the dataset group. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`
- `UPDATE_PENDING`, `UPDATE_IN_PROGRESS`, `UPDATE_FAILED`

The `UPDATE` states apply when you call the UpdateDatasetGroup (p. 188) operation.

> **Note**
> The `Status` of the dataset group must be `ACTIVE` before you can use the dataset group to create a predictor.

Type: String

Length Constraints: Maximum length of 256.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeDatasetImportJob

Service: Amazon Forecast Service

Describes a dataset import job created using the CreateDatasetImportJob (p. 120) operation.

In addition to listing the parameters provided in the `CreateDatasetImportJob` request, this operation includes the following properties:

- `CreationTime`
- `LastModificationTime`
- `DataSize`
- `FieldStatistics`
- `Status`
- `Message` - If an error occurred, information about the error.

Request Syntax

```
{
    "DatasetImportJobArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetImportJobArn (p. 154)

   The Amazon Resource Name (ARN) of the dataset import job.

   Type: String

   Length Constraints: Maximum length of 256.

   Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

   Required: Yes

Response Syntax

```
{
    "CreationTime": number,
    "DatasetArn": "string",
    "DatasetImportJobArn": "string",
    "DatasetImportJobName": "string",
    "DataSize": number,
    "DataSource": {
        "S3Config": {
            "KMSKeyArn": "string",
            "Path": "string",
            "RoleArn": "string"
        }
    },
    "FieldStatistics": {
        "string" : {
            "Avg": number,
            "Count": number,
            "CountDistinct": number,
```

```
        "CountNan": number,
        "CountNull": number,
        "Max": "string",
        "Min": "string",
        "Stddev": number
      }
    },
    "LastModificationTime": number,
    "Message": "string",
    "Status": "string",
    "TimestampFormat": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime (p. 154)

>   When the dataset import job was created.

>   Type: Timestamp

DatasetArn (p. 154)

>   The Amazon Resource Name (ARN) of the dataset that the training data was imported to.

>   Type: String

>   Length Constraints: Maximum length of 256.

>   Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

DatasetImportJobArn (p. 154)

>   The ARN of the dataset import job.

>   Type: String

>   Length Constraints: Maximum length of 256.

>   Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

DatasetImportJobName (p. 154)

>   The name of the dataset import job.

>   Type: String

>   Length Constraints: Minimum length of 1. Maximum length of 63.

>   Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

DataSize (p. 154)

>   The size of the dataset in gigabytes (GB) after the import job has finished.

>   Type: Double

DataSource (p. 154)

>   The location of the training data to import and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the data.

>   If encryption is used, `DataSource` includes an AWS Key Management Service (KMS) key.

Type: DataSource (p. 204) object

FieldStatistics (p. 154)

Statistical information about each field in the input data.

Type: String to Statistics (p. 232) object map

Key Length Constraints: Maximum length of 256.

Key Pattern: `^[a-zA-Z0-9\_]+$`

LastModificationTime (p. 154)

The last time that the dataset was modified. The time depends on the status of the job, as follows:

- `CREATE_PENDING` - The same time as `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Message (p. 154)

If an error occurred, an informational message about the error.

Type: String

Status (p. 154)

The status of the dataset import job. The status is reflected in the status of the dataset. For example, when the import job status is `CREATE_IN_PROGRESS`, the status of the dataset is `UPDATE_IN_PROGRESS`. States include:

- `ACTIVE`
- `CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED`
- `DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

TimestampFormat (p. 154)

The format of timestamps in the dataset. The format that you specify depends on the `DataFrequency` specified when the dataset was created. The following formats are supported

- "yyyy-MM-dd"

  For the following data frequencies: Y, M, W, and D
- "yyyy-MM-dd HH:mm:ss"

  For the following data frequencies: H, 30min, 15min, and 1min; and optionally, for: Y, M, W, and D

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\:\.\,\'\s]+$`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeForecast

Service: Amazon Forecast Service

Describes a forecast created using the CreateForecast (p. 123) operation.

In addition to listing the properties provided in the `CreateForecast` request, this operation lists the following properties:

- `DatasetGroupArn` - The dataset group that provided the training data.
- `CreationTime`
- `LastModificationTime`
- `Status`
- `Message` - If an error occurred, information about the error.

Request Syntax

```
{
    "ForecastArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastArn (p. 158)

> The Amazon Resource Name (ARN) of the forecast.
>
> Type: String
>
> Length Constraints: Maximum length of 256.
>
> Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`
>
> Required: Yes

Response Syntax

```
{
    "CreationTime": number,
    "DatasetGroupArn": "string",
    "ForecastArn": "string",
    "ForecastName": "string",
    "ForecastTypes": [ "string" ],
    "LastModificationTime": number,
    "Message": "string",
    "PredictorArn": "string",
    "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime (p. 158)

When the forecast creation task was created.

Type: Timestamp

DatasetGroupArn (p. 158)

The ARN of the dataset group that provided the data used to train the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

ForecastArn (p. 158)

The forecast ARN as specified in the request.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

ForecastName (p. 158)

The name of the forecast.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

ForecastTypes (p. 158)

The quantiles at which probabilistic forecasts were generated.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Pattern: `(^0?\.\d\d?$|^mean$)`

LastModificationTime (p. 158)

Initially, the same as `CreationTime` (status is `CREATE_PENDING`). Updated when inference (creating the forecast) starts (status changed to `CREATE_IN_PROGRESS`), and when inference is complete (status changed to `ACTIVE`) or fails (status changed to `CREATE_FAILED`).

Type: Timestamp

Message (p. 158)

If an error occurred, an informational message about the error.

Type: String

PredictorArn (p. 158)

The ARN of the predictor used to generate the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Status (p. 158)

The status of the forecast. States include:

- `ACTIVE`
- `CREATE_PENDING`, `CREATE_IN_PROGRESS`, `CREATE_FAILED`
- `DELETE_PENDING`, `DELETE_IN_PROGRESS`, `DELETE_FAILED`

   Note
   The `Status` of the forecast must be `ACTIVE` before you can query or export the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_]+$`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribeForecastExportJob

Service: Amazon Forecast Service

Describes a forecast export job created using the CreateForecastExportJob (p. 126) operation.

In addition to listing the properties provided by the user in the `CreateForecastExportJob` request, this operation lists the following properties:

- `CreationTime`
- `LastModificationTime`
- `Status`
- `Message` - If an error occurred, information about the error.

Request Syntax

```
{
    "ForecastExportJobArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

ForecastExportJobArn (p. 161)

    The Amazon Resource Name (ARN) of the forecast export job.

    Type: String

    Length Constraints: Maximum length of 256.

    Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

    Required: Yes

Response Syntax

```
{
    "CreationTime": number,
    "Destination": {
        "S3Config": {
            "KMSKeyArn": "string",
            "Path": "string",
            "RoleArn": "string"
        }
    },
    "ForecastArn": "string",
    "ForecastExportJobArn": "string",
    "ForecastExportJobName": "string",
    "LastModificationTime": number,
    "Message": "string",
    "Status": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

CreationTime (p. 161)

When the forecast export job was created.

Type: Timestamp

Destination (p. 161)

The path to the Amazon Simple Storage Service (Amazon S3) bucket where the forecast is exported.

Type: DataDestination (p. 198) object

ForecastArn (p. 161)

The Amazon Resource Name (ARN) of the exported forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

ForecastExportJobArn (p. 161)

The ARN of the forecast export job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

ForecastExportJobName (p. 161)

The name of the forecast export job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

LastModificationTime (p. 161)

When the last successful export job finished.

Type: Timestamp

Message (p. 161)

If an error occurred, an informational message about the error.

Type: String

Status (p. 161)

The status of the forecast export job. States include:

- `ACTIVE`
- `CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED`
- `DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED`

  Note
  The `Status` of the forecast export job must be `ACTIVE` before you can access the forecast in your S3 bucket.

Type: String

Length Constraints: Maximum length of 256.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DescribePredictor

Service: Amazon Forecast Service

Describes a predictor created using the CreatePredictor (p. 129) operation.

In addition to listing the properties provided in the `CreatePredictor` request, this operation lists the following properties:

- `DatasetImportJobArns` - The dataset import jobs used to import training data.
- `AutoMLAlgorithmArns` - If AutoML is performed, the algorithms that were evaluated.
- `CreationTime`
- `LastModificationTime`
- `Status`
- `Message` - If an error occurred, information about the error.

Request Syntax

```
{
    "PredictorArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

PredictorArn (p. 164)

The Amazon Resource Name (ARN) of the predictor that you want information about.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

Response Syntax

```
{
    "AlgorithmArn": "string",
    "AutoMLAlgorithmArns": [ "string" ],
    "CreationTime": number,
    "DatasetImportJobArns": [ "string" ],
    "EncryptionConfig": {
        "KMSKeyArn": "string",
        "RoleArn": "string"
    },
    "EvaluationParameters": {
        "BackTestWindowOffset": number,
        "NumberOfBacktestWindows": number
    },
    "FeaturizationConfig": {
        "Featurizations": [
            {
                "AttributeName": "string",
                "FeaturizationPipeline": [
```

```
                {
                    "FeaturizationMethodName": "string",
                    "FeaturizationMethodParameters": {
                        "string" : "string"
                    }
                }
            ]
        }
    ],
    "ForecastDimensions": [ "string" ],
    "ForecastFrequency": "string"
},
"ForecastHorizon": number,
"HPOConfig": {
    "ParameterRanges": {
        "CategoricalParameterRanges": [
            {
                "Name": "string",
                "Values": [ "string" ]
            }
        ],
        "ContinuousParameterRanges": [
            {
                "MaxValue": number,
                "MinValue": number,
                "Name": "string",
                "ScalingType": "string"
            }
        ],
        "IntegerParameterRanges": [
            {
                "MaxValue": number,
                "MinValue": number,
                "Name": "string",
                "ScalingType": "string"
            }
        ]
    }
},
"InputDataConfig": {
    "DatasetGroupArn": "string",
    "SupplementaryFeatures": [
        {
            "Name": "string",
            "Value": "string"
        }
    ]
},
"LastModificationTime": number,
"Message": "string",
"PerformAutoML": boolean,
"PerformHPO": boolean,
"PredictorArn": "string",
"PredictorExecutionDetails": {
    "PredictorExecutions": [
        {
            "AlgorithmArn": "string",
            "TestWindows": [
                {
                    "Message": "string",
                    "Status": "string",
                    "TestWindowEnd": number,
                    "TestWindowStart": number
                }
            ]
        }
```

```
        ]
    },
    "PredictorName": "string",
    "Status": "string",
    "TrainingParameters": {
        "string" : "string"
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

AlgorithmArn (p. 164)

The Amazon Resource Name (ARN) of the algorithm used for model training.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

AutoMLAlgorithmArns (p. 164)

When `PerformAutoML` is specified, the ARN of the chosen algorithm.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

CreationTime (p. 164)

When the model training task was created.

Type: Timestamp

DatasetImportJobArns (p. 164)

An array of the ARNs of the dataset import jobs used to import training data for the predictor.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

EncryptionConfig (p. 164)

An AWS Key Management Service (KMS) key and the AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key.

Type: EncryptionConfig (p. 205) object

EvaluationParameters (p. 164)

Used to override the default evaluation parameters of the specified algorithm. Amazon Forecast evaluates a predictor by splitting a dataset into training data and testing data. The evaluation parameters define how to perform the split and the number of iterations.

Type: EvaluationParameters (p. 206) object

FeaturizationConfig (p. 164)

The featurization configuration.

Type: FeaturizationConfig (p. 209) object

ForecastHorizon (p. 164)

The number of time-steps of the forecast. The forecast horizon is also called the prediction length.

Type: Integer

HPOConfig (p. 164)

The hyperparameter override values for the algorithm.

Type: HyperParameterTuningJobConfig (p. 218) object

InputDataConfig (p. 164)

Describes the dataset group that contains the data to use to train the predictor.

Type: InputDataConfig (p. 219) object

LastModificationTime (p. 164)

Initially, the same as `CreationTime` (when the status is `CREATE_PENDING`). This value is updated when training starts (when the status changes to `CREATE_IN_PROGRESS`), and when training has completed (when the status changes to `ACTIVE`) or fails (when the status changes to `CREATE_FAILED`).

Type: Timestamp

Message (p. 164)

If an error occurred, an informational message about the error.

Type: String

PerformAutoML (p. 164)

Whether the predictor is set to perform AutoML.

Type: Boolean

PerformHPO (p. 164)

Whether the predictor is set to perform hyperparameter optimization (HPO).

Type: Boolean

PredictorArn (p. 164)

The ARN of the predictor.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

PredictorExecutionDetails (p. 164)

Details on the the status and results of the backtests performed to evaluate the accuracy of the predictor. You specify the number of backtests to perform when you call the CreatePredictor (p. 129) operation.

Type: PredictorExecutionDetails (p. 225) object

PredictorName (p. 164)

The name of the predictor.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Status (p. 164)

The status of the predictor. States include:

- `ACTIVE`
- `CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED`
- `DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED`
- `UPDATE_PENDING, UPDATE_IN_PROGRESS, UPDATE_FAILED`

    Note
    The `Status` of the predictor must be `ACTIVE` before you can use the predictor to create a forecast.

Type: String

Length Constraints: Maximum length of 256.

TrainingParameters (p. 164)

The default training parameters or overrides selected during model training. If using the AutoML algorithm or if HPO is turned on while using the DeepAR+ algorithms, the optimized values for the chosen hyperparameters are returned. For more information, see Choosing an Amazon Forecast Algorithm (p. 55).

Type: String to string map

Key Length Constraints: Maximum length of 256.

Key Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]]\,\\]+$`

Value Length Constraints: Maximum length of 256.

Value Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]]\,\"\\\s]+$`

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetAccuracyMetrics

Service: Amazon Forecast Service

Provides metrics on the accuracy of the models that were trained by the CreatePredictor (p. 129) operation. Use metrics to see how well the model performed and to decide whether to use the predictor to generate a forecast. For more information, see Evaluating Predictor Accuracy (p. 67).

This operation generates metrics for each backtest window that was evaluated. The number of backtest windows (`NumberOfBacktestWindows`) is specified using the EvaluationParameters (p. 206) object, which is optionally included in the `CreatePredictor` request. If `NumberOfBacktestWindows` isn't specified, the number defaults to one.

The parameters of the `filling` method determine which items contribute to the metrics. If you want all items to contribute, specify `zero`. If you want only those items that have complete data in the range being evaluated to contribute, specify `nan`. For more information, see FeaturizationMethod (p. 211).

> **Note**
> Before you can get accuracy metrics, the `Status` of the predictor must be `ACTIVE`, signifying that training has completed. To get the status, use the DescribePredictor (p. 164) operation.

Request Syntax

```
{
    "PredictorArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

PredictorArn (p. 170)

The Amazon Resource Name (ARN) of the predictor to get metrics for.

Type: String

Length Constraints: Maximum length of 256.

Pattern: ^[a-zA-Z0-9\-_\.\/\:]+$

Required: Yes

Response Syntax

```
{
    "PredictorEvaluationResults": [
        {
            "AlgorithmArn": "string",
            "TestWindows": [
                {
                    "EvaluationType": "string",
                    "ItemCount": number,
                    "Metrics": {
                        "RMSE": number,
                        "WeightedQuantileLosses": [
                            {
                                "LossValue": number,
                                "Quantile": number
                            }
```

```
                ]
            },
            "TestWindowEnd": number,
            "TestWindowStart": number
        }
    ]
    }
    ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

PredictorEvaluationResults (p. 170)

An array of results from evaluating the predictor.

Type: Array of EvaluationResult (p. 207) objects

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListDatasetGroups

Service: Amazon Forecast Service

Returns a list of dataset groups created using the CreateDatasetGroup (p. 117) operation. For each dataset group, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). You can retrieve the complete set of properties by using the dataset group ARN with the DescribeDatasetGroup (p. 151) operation.

Request Syntax

```
{
    "MaxResults": number,
    "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

MaxResults (p. 172)

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken (p. 172)

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Required: No

Response Syntax

```
{
    "DatasetGroups": [
        {
            "CreationTime": number,
            "DatasetGroupArn": "string",
            "DatasetGroupName": "string",
            "LastModificationTime": number
        }
    ],
    "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DatasetGroups (p. 172)

An array of objects that summarize each dataset group's properties.

Type: Array of DatasetGroupSummary (p. 199) objects

NextToken (p. 172)

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Errors

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListDatasetImportJobs

Service: Amazon Forecast Service

Returns a list of dataset import jobs created using the CreateDatasetImportJob (p. 120) operation. For each import job, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). You can retrieve the complete set of properties by using the ARN with the DescribeDatasetImportJob (p. 154) operation. You can filter the list by providing an array of Filter (p. 213) objects.

Request Syntax

```
{
    "Filters": [
        {
            "Condition": "string",
            "Key": "string",
            "Value": "string"
        }
    ],
    "MaxResults": number,
    "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters (p. 174)

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the datasets that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the datasets that match the statement, specify `IS`. To exclude matching datasets, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `DatasetArn` and `Status`.
- `Value` - The value to match.

For example, to list all dataset import jobs whose status is ACTIVE, you specify the following filter:

`"Filters": [ { "Condition": "IS", "Key": "Status", "Value": "ACTIVE" } ]`

Type: Array of Filter (p. 213) objects

Required: No

MaxResults (p. 174)

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken (p. 174)

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Required: No

Response Syntax

```
{
    "DatasetImportJobs": [
        {
            "CreationTime": number,
            "DatasetImportJobArn": "string",
            "DatasetImportJobName": "string",
            "DataSource": {
                "S3Config": {
                    "KMSKeyArn": "string",
                    "Path": "string",
                    "RoleArn": "string"
                }
            },
            "LastModificationTime": number,
            "Message": "string",
            "Status": "string"
        }
    ],
    "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

DatasetImportJobs (p. 175)

An array of objects that summarize each dataset import job's properties.

Type: Array of DatasetImportJobSummary (p. 200) objects

NextToken (p. 175)

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListDatasets

Service: Amazon Forecast Service

Returns a list of datasets created using the CreateDataset (p. 113) operation. For each dataset, a summary of its properties, including its Amazon Resource Name (ARN), is returned. To retrieve the complete set of properties, use the ARN with the DescribeDataset (p. 147) operation.

Request Syntax

```
{
    "MaxResults": number,
    "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

MaxResults (p. 177)

>   The number of items to return in the response.

>   Type: Integer

>   Valid Range: Minimum value of 1. Maximum value of 100.

>   Required: No

NextToken (p. 177)

>   If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

>   Type: String

>   Length Constraints: Minimum length of 1. Maximum length of 3000.

>   Required: No

Response Syntax

```
{
    "Datasets": [
        {
            "CreationTime": number,
            "DatasetArn": "string",
            "DatasetName": "string",
            "DatasetType": "string",
            "Domain": "string",
            "LastModificationTime": number
        }
    ],
    "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Datasets (p. 177)

An array of objects that summarize each dataset's properties.

Type: Array of DatasetSummary (p. 202) objects

NextToken (p. 177)

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Errors

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListForecastExportJobs

Service: Amazon Forecast Service

Returns a list of forecast export jobs created using the CreateForecastExportJob (p. 126) operation. For each forecast export job, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). To retrieve the complete set of properties, use the ARN with the DescribeForecastExportJob (p. 161) operation. You can filter the list using an array of Filter (p. 213) objects.

Request Syntax

```
{
    "Filters": [
        {
            "Condition": "string",
            "Key": "string",
            "Value": "string"
        }
    ],
    "MaxResults": number,
    "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters (p. 179)

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the forecast export jobs that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the forecast export jobs that match the statement, specify `IS`. To exclude matching forecast export jobs, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `ForecastArn` and `Status`.
- `Value` - The value to match.

For example, to list all jobs that export a forecast named *electricityforecast*, specify the following filter:

```
"Filters": [ { "Condition": "IS", "Key": "ForecastArn", "Value":
"arn:aws:forecast:us-west-2:<acct-id>:forecast/electricityforecast" } ]
```

Type: Array of Filter (p. 213) objects

Required: No

MaxResults (p. 179)

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken (p. 179)

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Required: No

Response Syntax

```
{
    "ForecastExportJobs": [
        {
            "CreationTime": number,
            "Destination": {
                "S3Config": {
                    "KMSKeyArn": "string",
                    "Path": "string",
                    "RoleArn": "string"
                }
            },
            "ForecastExportJobArn": "string",
            "ForecastExportJobName": "string",
            "LastModificationTime": number,
            "Message": "string",
            "Status": "string"
        }
    ],
    "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ForecastExportJobs (p. 180)

An array of objects that summarize each export job's properties.

Type: Array of ForecastExportJobSummary (p. 214) objects

NextToken (p. 180)

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListForecasts

Service: Amazon Forecast Service

Returns a list of forecasts created using the CreateForecast (p. 123) operation. For each forecast, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). To retrieve the complete set of properties, specify the ARN with the DescribeForecast (p. 158) operation. You can filter the list using an array of Filter (p. 213) objects.

Request Syntax

```
{
    "Filters": [
        {
            "Condition": "string",
            "Key": "string",
            "Value": "string"
        }
    ],
    "MaxResults": number,
    "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters (p. 182)

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the forecasts that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the forecasts that match the statement, specify `IS`. To exclude matching forecasts, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `DatasetGroupArn`, `PredictorArn`, and `Status`.
- `Value` - The value to match.

For example, to list all forecasts whose status is not ACTIVE, you would specify:

`"Filters": [ { "Condition": "IS_NOT", "Key": "Status", "Value": "ACTIVE" } ]`

Type: Array of Filter (p. 213) objects

Required: No

MaxResults (p. 182)

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken (p. 182)

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Required: No

Response Syntax

```
{
    "Forecasts": [
        {
            "CreationTime": number,
            "DatasetGroupArn": "string",
            "ForecastArn": "string",
            "ForecastName": "string",
            "LastModificationTime": number,
            "Message": "string",
            "PredictorArn": "string",
            "Status": "string"
        }
    ],
    "NextToken": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Forecasts (p. 183)

An array of objects that summarize each forecast's properties.

Type: Array of ForecastSummary (p. 216) objects

NextToken (p. 183)

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListPredictors

Service: Amazon Forecast Service

Returns a list of predictors created using the CreatePredictor (p. 129) operation. For each predictor, this operation returns a summary of its properties, including its Amazon Resource Name (ARN). You can retrieve the complete set of properties by using the ARN with the DescribePredictor (p. 164) operation. You can filter the list using an array of Filter (p. 213) objects.

Request Syntax

```
{
    "Filters": [
        {
            "Condition": "string",
            "Key": "string",
            "Value": "string"
        }
    ],
    "MaxResults": number,
    "NextToken": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

Filters (p. 185)

An array of filters. For each filter, you provide a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the predictors that match the statement from the list, respectively. The match statement consists of a key and a value.

Filter properties

- `Condition` - The condition to apply. Valid values are `IS` and `IS_NOT`. To include the predictors that match the statement, specify `IS`. To exclude matching predictors, specify `IS_NOT`.
- `Key` - The name of the parameter to filter on. Valid values are `DatasetGroupArn` and `Status`.
- `Value` - The value to match.

For example, to list all predictors whose status is ACTIVE, you would specify:

`"Filters": [ { "Condition": "IS", "Key": "Status", "Value": "ACTIVE" } ]`

Type: Array of Filter (p. 213) objects

Required: No

MaxResults (p. 185)

The number of items to return in the response.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

NextToken (p. 185)

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Required: No

Response Syntax

```
{
    "NextToken": "string",
    "Predictors": [
        {
            "CreationTime": number,
            "DatasetGroupArn": "string",
            "LastModificationTime": number,
            "Message": "string",
            "PredictorArn": "string",
            "PredictorName": "string",
            "Status": "string"
        }
    ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken (p. 186)

If the response is truncated, Amazon Forecast returns this token. To retrieve the next set of results, use the token in the next request.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Predictors (p. 186)

An array of objects that summarize each predictor's properties.

Type: Array of PredictorSummary (p. 226) objects

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

UpdateDatasetGroup

Service: Amazon Forecast Service

Replaces the datasets in a dataset group with the specified datasets.

> **Note**
> The `Status` of the dataset group must be `ACTIVE` before you can use the dataset group to create a predictor. Use the DescribeDatasetGroup (p. 151) operation to get the status.

Request Syntax

```
{
    "DatasetArns": [ "string" ],
    "DatasetGroupArn": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

DatasetArns (p. 188)

An array of the Amazon Resource Names (ARNs) of the datasets to add to the dataset group.

Type: Array of strings

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

DatasetGroupArn (p. 188)

The ARN of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidInputException

We can't process the request because it includes an invalid value or a value that exceeds the valid range.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find a resource with that Amazon Resource Name (ARN). Check the ARN and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

Amazon Forecast Query Service

The following actions are supported by Amazon Forecast Query Service:

- QueryForecast (p. 190)

QueryForecast

Service: Amazon Forecast Query Service

Retrieves a forecast for a single item, filtered by the supplied criteria.

The criteria is a key-value pair. The key is either `item_id` (or the equivalent non-timestamp, non-target field) from the `TARGET_TIME_SERIES` dataset, or one of the forecast dimensions specified as part of the `FeaturizationConfig` object.

By default, `QueryForecast` returns the complete date range for the filtered forecast. You can request a specific date range.

To get the full forecast, use the CreateForecastExportJob operation.

> **Note**
> The forecasts generated by Amazon Forecast are in the same timezone as the dataset that was used to create the predictor.

Request Syntax

```
{
    "EndDate": "string",
    "Filters": {
        "string" : "string"
    },
    "ForecastArn": "string",
    "NextToken": "string",
    "StartDate": "string"
}
```

Request Parameters

The request accepts the following data in JSON format.

EndDate (p. 190)

The end date for the forecast. Specify the date using this format: yyyy-MM-dd'T'HH:mm:ss (ISO 8601 format). For example, 2015-01-01T20:00:00.

Type: String

Required: No

Filters (p. 190)

The filtering criteria to apply when retrieving the forecast. For example, to get the forecast for `client_21` in the electricity usage dataset, specify the following:

`{"item_id" : "client_21"}`

To get the full forecast, use the CreateForecastExportJob operation.

Type: String to string map

Key Length Constraints: Maximum length of 256.

Key Pattern: ^[a-zA-Z0-9_\-]+$

Value Length Constraints: Maximum length of 256.

Required: Yes

ForecastArn (p. 190)

The Amazon Resource Name (ARN) of the forecast to query.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:([a-z\d-]+):forecast:.*:.*:.+`

Required: Yes

NextToken (p. 190)

If the result of the previous request was truncated, the response includes a `NextToken`. To retrieve the next set of results, use the token in the next request. Tokens expire after 24 hours.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 3000.

Required: No

StartDate (p. 190)

The start date for the forecast. Specify the date using this format: yyyy-MM-dd'T'HH:mm:ss (ISO 8601 format). For example, 2015-01-01T08:00:00.

Type: String

Required: No

Response Syntax

```
{
    "Forecast": {
        "Predictions": {
            "string" : [
                {
                    "Timestamp": "string",
                    "Value": number
                }
            ]
        }
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Forecast (p. 191)

The forecast.

Type: Forecast (p. 241) object

Errors

InvalidInputException

The value is invalid or is too long.

HTTP Status Code: 400

InvalidNextTokenException

The token is not valid. Tokens expire after 24 hours.

HTTP Status Code: 400

LimitExceededException

The limit on the number of requests per second has been exceeded.

HTTP Status Code: 400

ResourceInUseException

The specified resource is in use.

HTTP Status Code: 400

ResourceNotFoundException

We can't find that resource. Check the information that you've provided and try again.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

Data Types

The following data types are supported by Amazon Forecast Service:

The following data types are supported by Amazon Forecast Query Service:

Amazon Forecast Service

The following data types are supported by Amazon Forecast Service:

CategoricalParameterRange

Service: Amazon Forecast Service

Specifies a categorical hyperparameter and it's range of tunable values. This object is part of the ParameterRanges (p. 223) object.

Contents

Name

The name of the categorical hyperparameter to tune.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Values

A list of the tunable categories for the hyperparameter.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_\-]+$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

ContinuousParameterRange

Service: Amazon Forecast Service

Specifies a continuous hyperparameter and it's range of tunable values. This object is part of the ParameterRanges (p. 223) object.

Contents

MaxValue

The maximum tunable value of the hyperparameter.

Type: Double

Required: Yes

MinValue

The minimum tunable value of the hyperparameter.

Type: Double

Required: Yes

Name

The name of the hyperparameter to tune.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

ScalingType

The scale that hyperparameter tuning uses to search the hyperparameter range. Valid values:
Auto

Amazon Forecast hyperparameter tuning chooses the best scale for the hyperparameter.
Linear

Hyperparameter tuning searches the values in the hyperparameter range by using a linear scale.
Logarithmic

Hyperparameter tuning searches the values in the hyperparameter range by using a logarithmic scale.

Logarithmic scaling works only for ranges that have values greater than 0.
ReverseLogarithmic

hyperparameter tuning searches the values in the hyperparameter range by using a reverse logarithmic scale.

Reverse logarithmic scaling works only for ranges that are entirely within the range 0 <= x < 1.0.

For information about choosing a hyperparameter scale, see Hyperparameter Scaling. One of the following values:

Type: String

Valid Values: `Auto | Linear | Logarithmic | ReverseLogarithmic`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

DataDestination

Service: Amazon Forecast Service

The destination for an exported forecast, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the location and, optionally, an AWS Key Management Service (KMS) key. This object is submitted in the CreateForecastExportJob (p. 126) request.

Contents

S3Config

The path to an Amazon Simple Storage Service (Amazon S3) bucket along with the credentials to access the bucket.

Type: S3Config (p. 228) object

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

DatasetGroupSummary

Service: Amazon Forecast Service

Provides a summary of the dataset group properties used in the ListDatasetGroups (p. 172) operation. To get the complete set of properties, call the DescribeDatasetGroup (p. 151) operation, and provide the `DatasetGroupArn`.

Contents

CreationTime

When the dataset group was created.

Type: Timestamp

Required: No

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

DatasetGroupName

The name of the dataset group.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

LastModificationTime

When the dataset group was created or last updated from a call to the UpdateDatasetGroup (p. 188) operation. While the dataset group is being updated, `LastModificationTime` is the current time of the `ListDatasetGroups` call.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

DatasetImportJobSummary

Service: Amazon Forecast Service

Provides a summary of the dataset import job properties used in the ListDatasetImportJobs (p. 174) operation. To get the complete set of properties, call the DescribeDatasetImportJob (p. 154) operation, and provide the `DatasetImportJobArn`.

Contents

CreationTime

When the dataset import job was created.

Type: Timestamp

Required: No

DatasetImportJobArn

The Amazon Resource Name (ARN) of the dataset import job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

DatasetImportJobName

The name of the dataset import job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

DataSource

The location of the training data to import and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the data. The training data must be stored in an Amazon S3 bucket.

If encryption is used, `DataSource` includes an AWS Key Management Service (KMS) key.

Type: DataSource (p. 204) object

Required: No

LastModificationTime

The last time that the dataset was modified. The time depends on the status of the job, as follows:

- `CREATE_PENDING` - The same time as `CreationTime`.
- `CREATE_IN_PROGRESS` - The current timestamp.
- `ACTIVE` or `CREATE_FAILED` - When the job finished or failed.

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

Status

The status of the dataset import job. The status is reflected in the status of the dataset. For example, when the import job status is `CREATE_IN_PROGRESS`, the status of the dataset is `UPDATE_IN_PROGRESS`. States include:

- `ACTIVE`
- `CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED`
- `DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

DatasetSummary

Service: Amazon Forecast Service

Provides a summary of the dataset properties used in the ListDatasets (p. 177) operation. To get the complete set of properties, call the DescribeDataset (p. 147) operation, and provide the `DatasetArn`.

Contents

CreationTime

When the dataset was created.

Type: Timestamp

Required: No

DatasetArn

The Amazon Resource Name (ARN) of the dataset.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

DatasetName

The name of the dataset.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

DatasetType

The dataset type.

Type: String

Valid Values: `TARGET_TIME_SERIES` | `RELATED_TIME_SERIES` | `ITEM_METADATA`

Required: No

Domain

The domain associated with the dataset.

Type: String

Valid Values: `RETAIL` | `CUSTOM` | `INVENTORY_PLANNING` | `EC2_CAPACITY` | `WORK_FORCE` | `WEB_TRAFFIC` | `METRICS`

Required: No

LastModificationTime

When you create a dataset, `LastModificationTime` is the same as `CreationTime`. While data is being imported to the dataset, `LastModificationTime` is the current time of the `ListDatasets`

call. After a CreateDatasetImportJob (p. 120) operation has finished, `LastModificationTime` is when the import job completed or failed.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

DataSource

Service: Amazon Forecast Service

The source of your training data, an AWS Identity and Access Management (IAM) role that allows Amazon Forecast to access the data and, optionally, an AWS Key Management Service (KMS) key. This object is submitted in the CreateDatasetImportJob (p. 120) request.

Contents

S3Config

The path to the training data stored in an Amazon Simple Storage Service (Amazon S3) bucket along with the credentials to access the data.

Type: S3Config (p. 228) object

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

EncryptionConfig

Service: Amazon Forecast Service

An AWS Key Management Service (KMS) key and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the key. You can specify this optional object in the and requests.

Contents

KMSKeyArn

The Amazon Resource Name (ARN) of the KMS key.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:aws:kms:.*:key/.*`

Required: Yes

RoleArn

The ARN of the IAM role that Amazon Forecast can assume to access the AWS KMS key.

Passing a role across AWS accounts is not allowed. If you pass a role that isn't in your account, you get an `InvalidInputException` error.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

EvaluationParameters

Service: Amazon Forecast Service

Parameters that define how to split a dataset into training data and testing data, and the number of iterations to perform. These parameters are specified in the predefined algorithms but you can override them in the CreatePredictor (p. 129) request.

Contents

BackTestWindowOffset

The point from the end of the dataset where you want to split the data for model training and testing (evaluation). Specify the value as the number of data points. The default is the value of the forecast horizon. `BackTestWindowOffset` can be used to mimic a past virtual forecast start date. This value must be greater than or equal to the forecast horizon and less than half of the TARGET_TIME_SERIES dataset length.

`ForecastHorizon <= BackTestWindowOffset` < 1/2 * TARGET_TIME_SERIES dataset length

Type: Integer

Required: No

NumberOfBacktestWindows

The number of times to split the input data. The default is 1. Valid values are 1 through 5.

Type: Integer

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

EvaluationResult

Service: Amazon Forecast Service

The results of evaluating an algorithm. Returned as part of the GetAccuracyMetrics (p. 170) response.

Contents

AlgorithmArn

The Amazon Resource Name (ARN) of the algorithm that was evaluated.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

TestWindows

The array of test windows used for evaluating the algorithm. The `NumberOfBacktestWindows` from the EvaluationParameters (p. 206) object determines the number of windows in the array.

Type: Array of WindowSummary (p. 238) objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

Featurization

Service: Amazon Forecast Service

Provides featurization (transformation) information for a dataset field. This object is part of the FeaturizationConfig (p. 209) object.

For example:

{

"AttributeName": "demand",

FeaturizationPipeline [{

"FeaturizationMethodName": "filling",

"FeaturizationMethodParameters": {"aggregation": "avg", "backfill": "nan"}

}]

}

Contents

AttributeName

The name of the schema attribute that specifies the data field to be featurized. Only the `target` field of the `TARGET_TIME_SERIES` dataset type is supported. For example, for the `RETAIL` domain, the target is `demand`, and for the `CUSTOM` domain, the target is `target_value`.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

FeaturizationPipeline

An array of one `FeaturizationMethod` object that specifies the feature transformation method.

Type: Array of FeaturizationMethod (p. 211) objects

Array Members: Fixed number of 1 item.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

FeaturizationConfig

Service: Amazon Forecast Service

In a CreatePredictor (p. 129) operation, the specified algorithm trains a model using the specified dataset group. You can optionally tell the operation to modify data fields prior to training a model. These modifications are referred to as *featurization*.

You define featurization using the `FeaturizationConfig` object. You specify an array of transformations, one for each field that you want to featurize. You then include the `FeaturizationConfig` object in your `CreatePredictor` request. Amazon Forecast applies the featurization to the `TARGET_TIME_SERIES` dataset before model training.

You can create multiple featurization configurations. For example, you might call the `CreatePredictor` operation twice by specifying different featurization configurations.

Contents

Featurizations

An array of featurization (transformation) information for the fields of a dataset. Only a single featurization is supported.

Type: Array of Featurization (p. 208) objects

Array Members: Fixed number of 1 item.

Required: No

ForecastDimensions

An array of dimension (field) names that specify how to group the generated forecast.

For example, suppose that you are generating a forecast for item sales across all of your stores, and your dataset contains a `store_id` field. If you want the sales forecast for each item by store, you would specify `store_id` as the dimension.

All forecast dimensions specified in the `TARGET_TIME_SERIES` dataset don't need to be specified in the `CreatePredictor` request. All forecast dimensions specified in the `RELATED_TIME_SERIES` dataset must be specified in the `CreatePredictor` request.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 5 items.

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

ForecastFrequency

The frequency of predictions in a forecast.

Valid intervals are Y (Year), M (Month), W (Week), D (Day), H (Hour), 30min (30 minutes), 15min (15 minutes), 10min (10 minutes), 5min (5 minutes), and 1min (1 minute). For example, "Y" indicates every year and "5min" indicates every five minutes.

The frequency must be greater than or equal to the TARGET_TIME_SERIES dataset frequency.

When a RELATED_TIME_SERIES dataset is provided, the frequency must be equal to the RELATED_TIME_SERIES dataset frequency.

Type: String

Pattern: `^Y|M|W|D|H|30min|15min|10min|5min|1min$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

FeaturizationMethod

Service: Amazon Forecast Service

Provides information about the method that featurizes (transforms) a dataset field. The method is part of the `FeaturizationPipeline` of the Featurization (p. 208) object. If you don't specify `FeaturizationMethodParameters`, Amazon Forecast uses default parameters.

The following is an example of how you specify a `FeaturizationMethod` object.

```
{

"FeaturizationMethodName": "filling",

"FeaturizationMethodParameters": {"aggregation": "avg", "backfill": "nan"}

}
```

Contents

FeaturizationMethodName

The name of the method. The "filling" method is the only supported method.

Type: String

Valid Values: `filling`

Required: Yes

FeaturizationMethodParameters

The method parameters (key-value pairs). Specify these parameters to override the default values. The following list shows the parameters and their valid values. Bold signifies the default value.

- `aggregation`: **sum**, `avg`, `first`, `min`, `max`
- `frontfill`: **none**
- `middlefill`: **zero**, `nan` (not a number)
- `backfill`: **zero**, `nan`

Type: String to string map

Key Length Constraints: Maximum length of 256.

Key Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]]\,\\]+$`

Value Length Constraints: Maximum length of 256.

Value Pattern: `^[a-zA-Z0-9\-\_\.\/\[\]]\,\"\\\s]+$`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

Filter

Service: Amazon Forecast Service

Describes a filter for choosing a subset of objects. Each filter consists of a condition and a match statement. The condition is either `IS` or `IS_NOT`, which specifies whether to include or exclude the objects that match the statement, respectively. The match statement consists of a key and a value.

Contents

Condition

The condition to apply. To include the objects that match the statement, specify `IS`. To exclude matching objects, specify `IS_NOT`.

Type: String

Valid Values: `IS | IS_NOT`

Required: Yes

Key

The name of the parameter to filter on.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_]+$`

Required: Yes

Value

The value to match.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

ForecastExportJobSummary

Service: Amazon Forecast Service

Provides a summary of the forecast export job properties used in the ListForecastExportJobs (p. 179) operation. To get the complete set of properties, call the DescribeForecastExportJob (p. 161) operation, and provide the listed `ForecastExportJobArn`.

Contents

CreationTime

When the forecast export job was created.

Type: Timestamp

Required: No

Destination

The path to the Amazon Simple Storage Service (Amazon S3) bucket where the forecast is exported.

Type: DataDestination (p. 198) object

Required: No

ForecastExportJobArn

The Amazon Resource Name (ARN) of the forecast export job.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

ForecastExportJobName

The name of the forecast export job.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

LastModificationTime

When the last successful export job finished.

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

Status

The status of the forecast export job. States include:

- `ACTIVE`
- `CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED`
- `DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED`

> **Note**
> The `Status` of the forecast export job must be `ACTIVE` before you can access the forecast in your S3 bucket.

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

ForecastSummary

Service: Amazon Forecast Service

Provides a summary of the forecast properties used in the ListForecasts (p. 182) operation. To get the complete set of properties, call the DescribeForecast (p. 158) operation, and provide the `ForecastArn` that is listed in the summary.

Contents

CreationTime

When the forecast creation task was created.

Type: Timestamp

Required: No

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group that provided the data used to train the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_]+$`

Required: No

ForecastArn

The ARN of the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

ForecastName

The name of the forecast.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

LastModificationTime

Initially, the same as `CreationTime` (status is `CREATE_PENDING`). Updated when inference (creating the forecast) starts (status changed to `CREATE_IN_PROGRESS`), and when inference is complete (status changed to `ACTIVE`) or fails (status changed to `CREATE_FAILED`).

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

PredictorArn

The ARN of the predictor used to generate the forecast.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_]+$`

Required: No

Status

The status of the forecast. States include:

- `ACTIVE`
- `CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED`
- `DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED`

  Note
  The `Status` of the forecast must be `ACTIVE` before you can query or export the forecast.

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

HyperParameterTuningJobConfig

Service: Amazon Forecast Service

Configuration information for a hyperparameter tuning job. You specify this object in the
CreatePredictor (p. 129) request.

A *hyperparameter* is a parameter that governs the model training process. You set hyperparameters
before training starts, unlike model parameters, which are determined during training. The values of the
hyperparameters effect which values are chosen for the model parameters.

In a *hyperparameter tuning job*, Amazon Forecast chooses the set of hyperparameter values that
optimize a specified metric. Forecast accomplishes this by running many training jobs over a range of
hyperparameter values. The optimum set of values depends on the algorithm, the training data, and the
specified metric objective.

Contents

ParameterRanges

Specifies the ranges of valid values for the hyperparameters.

Type: ParameterRanges (p. 223) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

InputDataConfig

Service: Amazon Forecast Service

The data used to train a predictor. The data includes a dataset group and any supplementary features. You specify this object in the CreatePredictor (p. 129) request.

Contents

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

SupplementaryFeatures

An array of supplementary features. The only supported feature is a holiday calendar.

Type: Array of SupplementaryFeature (p. 234) objects

Array Members: Fixed number of 1 item.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

IntegerParameterRange

Service: Amazon Forecast Service

Specifies an integer hyperparameter and it's range of tunable values. This object is part of the ParameterRanges (p. 223) object.

Contents

MaxValue

The maximum tunable value of the hyperparameter.

Type: Integer

Required: Yes

MinValue

The minimum tunable value of the hyperparameter.

Type: Integer

Required: Yes

Name

The name of the hyperparameter to tune.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

ScalingType

The scale that hyperparameter tuning uses to search the hyperparameter range. Valid values:
Auto

Amazon Forecast hyperparameter tuning chooses the best scale for the hyperparameter.
Linear

Hyperparameter tuning searches the values in the hyperparameter range by using a linear scale.
Logarithmic

Hyperparameter tuning searches the values in the hyperparameter range by using a logarithmic scale.

Logarithmic scaling works only for ranges that have values greater than 0.
ReverseLogarithmic

Not supported for `IntegerParameterRange`.

Reverse logarithmic scaling works only for ranges that are entirely within the range 0 <= x < 1.0.

For information about choosing a hyperparameter scale, see Hyperparameter Scaling. One of the following values:

Type: String

Valid Values: `Auto | Linear | Logarithmic | ReverseLogarithmic`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

Metrics

Service: Amazon Forecast Service

Provides metrics that are used to evaluate the performance of a predictor. This object is part of the WindowSummary (p. 238) object.

Contents

RMSE

The root mean square error (RMSE).

Type: Double

Required: No

WeightedQuantileLosses

An array of weighted quantile losses. Quantiles divide a probability distribution into regions of equal probability. The distribution in this case is the loss function.

Type: Array of WeightedQuantileLoss (p. 237) objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

ParameterRanges

Service: Amazon Forecast Service

Specifies the categorical, continuous, and integer hyperparameters, and their ranges of tunable values. The range of tunable values determines which values that a hyperparameter tuning job can choose for the specified hyperparameter. This object is part of the HyperParameterTuningJobConfig (p. 218) object.

Contents

CategoricalParameterRanges

Specifies the tunable range for each categorical hyperparameter.

Type: Array of CategoricalParameterRange (p. 195) objects

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Required: No

ContinuousParameterRanges

Specifies the tunable range for each continuous hyperparameter.

Type: Array of ContinuousParameterRange (p. 196) objects

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Required: No

IntegerParameterRanges

Specifies the tunable range for each integer hyperparameter.

Type: Array of IntegerParameterRange (p. 220) objects

Array Members: Minimum number of 1 item. Maximum number of 20 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

PredictorExecution

Service: Amazon Forecast Service

The algorithm used to perform a backtest and the status of those tests.

Contents

AlgorithmArn

The ARN of the algorithm used to test the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

TestWindows

An array of test windows used to evaluate the algorithm. The `NumberOfBacktestWindows` from the EvaluationParameters (p. 206) object determines the number of windows in the array.

Type: Array of TestWindowSummary (p. 236) objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

PredictorExecutionDetails

Service: Amazon Forecast Service

Contains details on the backtests performed to evaluate the accuracy of the predictor. The tests are returned in descending order of accuracy, with the most accurate backtest appearing first. You specify the number of backtests to perform when you call the CreatePredictor (p. 129) operation.

Contents

PredictorExecutions

An array of the backtests performed to evaluate the accuracy of the predictor against a particular algorithm. The `NumberOfBacktestWindows` from the EvaluationParameters (p. 206) object determines the number of windows in the array.

Type: Array of PredictorExecution (p. 224) objects

Array Members: Minimum number of 1 item. Maximum number of 5 items.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

PredictorSummary

Service: Amazon Forecast Service

Provides a summary of the predictor properties that are used in the ListPredictors (p. 185) operation. To get the complete set of properties, call the DescribePredictor (p. 164) operation, and provide the listed `PredictorArn`.

Contents

CreationTime

When the model training task was created.

Type: Timestamp

Required: No

DatasetGroupArn

The Amazon Resource Name (ARN) of the dataset group that contains the data used to train the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

LastModificationTime

Initially, the same as `CreationTime` (status is `CREATE_PENDING`). Updated when training starts (status changed to `CREATE_IN_PROGRESS`), and when training is complete (status changed to `ACTIVE`) or fails (status changed to `CREATE_FAILED`).

Type: Timestamp

Required: No

Message

If an error occurred, an informational message about the error.

Type: String

Required: No

PredictorArn

The ARN of the predictor.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: No

PredictorName

The name of the predictor.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

Status

The status of the predictor. States include:

- `ACTIVE`
- `CREATE_PENDING, CREATE_IN_PROGRESS, CREATE_FAILED`
- `DELETE_PENDING, DELETE_IN_PROGRESS, DELETE_FAILED`
- `UPDATE_PENDING, UPDATE_IN_PROGRESS, UPDATE_FAILED`

   Note
   The `Status` of the predictor must be `ACTIVE` before you can use the predictor to create a forecast.

Type: String

Length Constraints: Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

S3Config

Service: Amazon Forecast Service

The path to the file(s) in an Amazon Simple Storage Service (Amazon S3) bucket, and an AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the file(s). Optionally, includes an AWS Key Management Service (KMS) key. This object is part of the DataSource (p. 204) object that is submitted in the CreateDatasetImportJob (p. 120) request, and part of the DataDestination (p. 198) object that is submitted in the CreateForecastExportJob (p. 126) request.

Contents

KMSKeyArn

The Amazon Resource Name (ARN) of an AWS Key Management Service (KMS) key.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `arn:aws:kms:.*:key/.*`

Required: No

Path

The path to an Amazon Simple Storage Service (Amazon S3) bucket or file(s) in an Amazon S3 bucket.

Type: String

Pattern: `^s3://[a-z0-9].+$`

Required: Yes

RoleArn

The ARN of the AWS Identity and Access Management (IAM) role that Amazon Forecast can assume to access the Amazon S3 bucket or files. If you provide a value for the `KMSKeyArn` key, the role must allow access to the key.

Passing a role across AWS accounts is not allowed. If you pass a role that isn't in your account, you get an `InvalidInputException` error.

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\-\_\.\/\:]+$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

Schema

Service: Amazon Forecast Service

Defines the fields of a dataset. You specify this object in the CreateDataset (p. 113) request.

Contents

Attributes

An array of attributes specifying the name and type of each field in a dataset.

Type: Array of SchemaAttribute (p. 231) objects

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

SchemaAttribute

Service: Amazon Forecast Service

An attribute of a schema, which defines a dataset field. A schema attribute is required for every field in a dataset. The Schema (p. 230) object contains an array of `SchemaAttribute` objects.

Contents

AttributeName

The name of the dataset field.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: No

AttributeType

The data type of the field.

Type: String

Valid Values: `string | integer | float | timestamp`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

Statistics

Service: Amazon Forecast Service

Provides statistics for each data field imported into to an Amazon Forecast dataset with the
CreateDatasetImportJob (p. 120) operation.

Contents

Avg

> For a numeric field, the average value in the field.
>
> Type: Double
>
> Required: No

Count

> The number of values in the field.
>
> Type: Integer
>
> Required: No

CountDistinct

> The number of distinct values in the field.
>
> Type: Integer
>
> Required: No

CountNan

> The number of NAN (not a number) values in the field.
>
> Type: Integer
>
> Required: No

CountNull

> The number of null values in the field.
>
> Type: Integer
>
> Required: No

Max

> For a numeric field, the maximum value in the field.
>
> Type: String
>
> Length Constraints: Maximum length of 256.
>
> Pattern: `^[a-zA-Z0-9\_]+$`
>
> Required: No

Min

> For a numeric field, the minimum value in the field.
>
> Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_]+$`

Required: No

Stddev

For a numeric field, the standard deviation.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

SupplementaryFeature

Service: Amazon Forecast Service

Describes a supplementary feature of a dataset group. This object is part of the InputDataConfig (p. 219) object.

The only supported feature is a holiday calendar. If you use the calendar, all data in the datasets should belong to the same country as the calendar. For the holiday calendar data, see the Jollyday web site.

India and Korea's holidays are not included in the Jollyday library, but both are supported by Amazon Forecast. Their holidays are:

"IN" - INDIA

- `JANUARY 26 – REPUBLIC DAY`
- `AUGUST 15 – INDEPENDENCE DAY`
- `OCTOBER 2 GANDHI'S BIRTHDAY`

"KR" - KOREA

- `JANUARY 1 – NEW YEAR`
- `MARCH 1 – INDEPENDENCE MOVEMENT DAY`
- `MAY 5 – CHILDREN'S DAY`
- `JUNE 6 – MEMORIAL DAY`
- `AUGUST 15 – LIBERATION DAY`
- `OCTOBER 3 – NATIONAL FOUNDATION DAY`
- `OCTOBER 9 – HANGEUL DAY`
- `DECEMBER 25 – CHRISTMAS DAY`

Contents

Name

The name of the feature. This must be "holiday".

Type: String

Length Constraints: Minimum length of 1. Maximum length of 63.

Pattern: `^[a-zA-Z][a-zA-Z0-9_]*`

Required: Yes

Value

One of the following 2 letter country codes:
- "AR" - ARGENTINA
- "AT" - AUSTRIA
- "AU" - AUSTRALIA
- "BE" - BELGIUM
- "BR" - BRAZIL
- "CA" - CANADA
- "CN" - CHINA
- "CZ" - CZECH REPUBLIC

- "DK" - DENMARK
- "EC" - ECUADOR
- "FI" - FINLAND
- "FR" - FRANCE
- "DE" - GERMANY
- "HU" - HUNGARY
- "IE" - IRELAND
- "IN" - INDIA
- "IT" - ITALY
- "JP" - JAPAN
- "KR" - KOREA
- "LU" - LUXEMBOURG
- "MX" - MEXICO
- "NL" - NETHERLANDS
- "NO" - NORWAY
- "PL" - POLAND
- "PT" - PORTUGAL
- "RU" - RUSSIA
- "ZA" - SOUTH AFRICA
- "ES" - SPAIN
- "SE" - SWEDEN
- "CH" - SWITZERLAND
- "US" - UNITED STATES
- "UK" - UNITED KINGDOM

Type: String

Length Constraints: Maximum length of 256.

Pattern: `^[a-zA-Z0-9\_\-]+$`

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

TestWindowSummary

Service: Amazon Forecast Service

The status, start time, and end time of a backtest, as well as a failure reason if applicable.

Contents

Message

If the test failed, the reason why it failed.

Type: String

Required: No

Status

The status of the test. Possible status values are:

- `ACTIVE`
- `CREATE_IN_PROGRESS`
- `CREATE_FAILED`

Type: String

Length Constraints: Maximum length of 256.

Required: No

TestWindowEnd

The time at which the test ended.

Type: Timestamp

Required: No

TestWindowStart

The time at which the test began.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

WeightedQuantileLoss

Service: Amazon Forecast Service

The weighted loss value for a quantile. This object is part of the Metrics (p. 222) object.

Contents

LossValue

The difference between the predicted value and the actual value over the quantile, weighted (normalized) by dividing by the sum over all quantiles.

Type: Double

Required: No

Quantile

The quantile. Quantiles divide a probability distribution into regions of equal probability. For example, if the distribution was divided into 5 regions of equal probability, the quantiles would be 0.2, 0.4, 0.6, and 0.8.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

WindowSummary

Service: Amazon Forecast Service

The metrics for a time range within the evaluation portion of a dataset. This object is part of the EvaluationResult (p. 207) object.

The `TestWindowStart` and `TestWindowEnd` parameters are determined by the `BackTestWindowOffset` parameter of the EvaluationParameters (p. 206) object.

Contents

EvaluationType

The type of evaluation.
- `SUMMARY` - The average metrics across all windows.
- `COMPUTED` - The metrics for the specified window.

Type: String

Valid Values: `SUMMARY | COMPUTED`

Required: No

ItemCount

The number of data points within the window.

Type: Integer

Required: No

Metrics

Provides metrics used to evaluate the performance of a predictor.

Type: Metrics (p. 222) object

Required: No

TestWindowEnd

The timestamp that defines the end of the window.

Type: Timestamp

Required: No

TestWindowStart

The timestamp that defines the start of the window.

Type: Timestamp

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go

- AWS SDK for Java
- AWS SDK for Ruby V3

Amazon Forecast Query Service

The following data types are supported by Amazon Forecast Query Service:

- DataPoint (p. 240)
- Forecast (p. 241)

DataPoint

Service: Amazon Forecast Query Service

The forecast value for a specific date. Part of the Forecast (p. 241) object.

Contents

Timestamp

The timestamp of the specific forecast.

Type: String

Required: No

Value

The forecast value.

Type: Double

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

Forecast

Service: Amazon Forecast Query Service

Provides information about a forecast. Returned as part of the QueryForecast (p. 190) response.

Contents

Predictions

The forecast.

The *string* of the string-to-array map is one of the following values:

- p10
- p50
- p90

Type: String to array of DataPoint (p. 240) objects map

Key Length Constraints: Maximum length of 4.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V3

Common Errors

This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

IncompleteSignature

The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure

The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction

The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

InvalidClientTokenId

The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

InvalidParameterCombination

Parameters that must not be used together were used together.

HTTP Status Code: 400

InvalidParameterValue

An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

InvalidQueryParameter

The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

MalformedQueryString

The query string contains a syntax error.

HTTP Status Code: 404

MissingAction

The request is missing an action or a required parameter.

HTTP Status Code: 400

MissingAuthenticationToken

The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

MissingParameter

A required parameter for the specified action is not supplied.

HTTP Status Code: 400

OptInRequired

The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired

The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable

The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationError

The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400

Common Parameters

The following list contains the parameters that all actions use for signing Signature Version 4 requests with a query string. Any action-specific parameters are listed in the topic for that action. For more information about Signature Version 4, see Signature Version 4 Signing Process in the *Amazon Web Services General Reference*.

Action

The action to be performed.

Type: string

Required: Yes

Version

The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

X-Amz-Algorithm

The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: `AWS4-HMAC-SHA256`

Required: Conditional

X-Amz-Credential

The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string ("aws4_request"). The value is expressed in the following format: *access_key*/*YYYYMMDD*/*region*/*service*/aws4_request.

For more information, see Task 2: Create a String to Sign for Signature Version 4 in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-Date

The date that is used to create the signature. The format must be ISO 8601 basic format (YYYYMMDD'T'HHMMSS'Z'). For example, the following date time is a valid X-Amz-Date value: `20120325T120000Z`.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see Handling Dates in Signature Version 4 in the *Amazon Web Services General Reference*.

Type: string

Required: Conditional

X-Amz-Security-Token

The temporary security token that was obtained through a call to AWS Security Token Service (AWS STS). For a list of services that support temporary security credentials from AWS Security Token Service, go to AWS Services That Work with IAM in the *IAM User Guide*.

Condition: If you're using temporary security credentials from the AWS Security Token Service, you must include the security token.

Type: string

Required: Conditional

X-Amz-Signature

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-SignedHeaders

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see Task 1: Create a Canonical Request For Signature Version 4 in the *Amazon Web Services General Reference*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

Document History for Amazon Forecast

The following table describes important changes to the *Amazon Forecast Developer Guide*. For notifications about documentation updates, you can subscribe to the RSS feed.

- **Latest documentation update:** March 17, 2020

update-history-change	update-history-description	update-history-date
New Region (p. 245)	Amazon Forecast adds support for the Asia Pacific (Seoul), Asia Pacific (Mumbai), and Europe (Frankfurt) Regions. For a complete list of the AWS Regions supported by Amazon Forecast, see the AWS Region Table or AWS Regions and Endpoints in the *Amazon Web Services General Reference*.	March 17, 2020
New Region (p. 245)	Amazon Forecast adds support for the Asia Pacific (Seoul) Region. For a complete list of the AWS Regions supported by Amazon Forecast, see the AWS Region Table or AWS Regions and Endpoints in the *Amazon Web Services General Reference*.	January 27, 2020
New feature (p. 245)	Forecast now supports the ability to specify forecast quantiles. For more information, see CreateForecast in the Forecast API Guide.	November 22, 2019
Amazon Forecast general availabilty (p. 245)	Amazon Forecast is now available for general use.	August 21, 2019
Amazon Forecast preview release (p. 245)	This is the first preview release of the documentation for Amazon Forecast.	November 28, 2018

AWS glossary

For the latest AWS terminology, see the AWS glossary in the *AWS General Reference*.