Apama Predictive Analytics Plug-in

Version 10.7

October 2020

APAMA

This document applies to Apama Predictive Analytics Plug-in 10.7 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: PAD-UG-107-20201015

Table of Contents

About this Guide

This guide describes how to install and configure the Apama Predictive Analytics Plug-in.

Documentation roadmap

Apama Predictive Analytics Plug-in provides documentation in the following formats:

- HTML (viewable in a web browser)
- PDF (available from the documentation website)

You can access the HTML documentation on your machine after Predictive Analytics Plug-in has been installed:

- **Windows.** Select **Start > All Programs > Software AG > Tools > Apama Predictive Analytics Plug-in *n.n* > Apama Predictive Analytics Plug-in Documentation *n.n***. Note that **Software AG** is the default group name that can be changed during the installation.

Predictive Analytics Plug-in also provides the following API reference information:

- API Reference for Apama Predictive Analytics Plug-in EPL (Apamadoc)
- API Reference for Predictive Analytics Engine (Javadoc) - available only in the installed product

Online Information and Support

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at http://documentation.softwareag.com.

Software AG Empower Product Support Website

If you do not yet have an account for Empower, send an email to empower@softwareag.com with your name, company, and company email address and request an account.

Once you have an account, you can open Support Incidents online via the eService section of Empower at https://empower.softwareag.com/.

You can find product information on the Software AG Empower Product Support website at https://empower.softwareag.com.

To submit feature/enhancement requests, get information about product availability, and download products, go to Products.

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the Knowledge Center.

If you have any questions, you can find a local or toll-free number for your country in our Global Support Contact Directory at https://empower.softwareag.com/public_directory.aspx and give us a call.

Software AG TECHcommunity

You can find documentation and other technical information on the Software AG TECHcommunity website at http://techcommunity.softwareag.com. You can:

- Access product documentation, if you have TECHcommunity credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

1 Release Notes

Release Notes describes the changes introduced with the current Apama Predictive Analytics Plug-in release as well as earlier releases.

What's new in 10.7 release

This version of Apama Predictive Analytics Plug-in does not contain any new functionality.

What's new in 10.5 release

This version of Apama Predictive Analytics Plug-in does not contain any new functionality.

What's new in 10.3.1 release

This version of Apama Predictive Analytics Plug-in does not contain any new functionality.

What's new in 10.3.0 release

This version of Apama Predictive Analytics Plug-in does not contain any new functionality.

What's new in 10.2 release

- The Zementis license file should now be copied to *SAG_HOME*`/Zementis/adapa-bundle/lib`, where *SAG_HOME* is the root directory of your Software AG installation.

What's new in 10.1 release

- Predictive Analytics Engine has been updated to Zementis Predictive Analytics 10.1.

  Starting with Zementis Predictive Analytics 10.1, the `adapa-bundle` path will not contain a version number in the library name. That is, the library name will now be `adapa-bundle.jar` instead of `adapa-bundle-<version>.jar`.

  Projects dependent on ant macros (`predictive-analytics-support-macros.xml`) should no longer depend on *ADAPA_LIB_VER*.

- Added support for activating and deactivating models in Predictive Analytics plug-in at runtime.

- Any errors occurred during scoring of input requests are now forwarded using `PredictiveAnalytics_Error` instead of the earlier `ADAPA_Error`.

- Warnings occurred during scoring of input requests are now forwarded using `PredictiveAnalytics_Warning<N>` instead of the earlier `ADAPA_Warning_<N>`.

What's new in 10.0 release

- Predictive Analytics Engine has been updated to Zementis Predictive Analytics 5.0.1. If you have an existing installation, you must back up the artifacts uploaded in Predictive Analytics

Engine before you upgrade. Upgrading to Zementis Predictive Analytics 5.0.1 requires a clean repository store. The following features are available with this upgrade :

- Compatibility and automatic model conversion to PMML 4.3
- Support for new algorithms: Convolutional Neural Networks and k-Nearest Neighbor

- Added support for adding and updating models received as byte stream to Predictive Analytics plug-in.

What's new in 9.12 release

- Predictive Analytics Engine has been migrated from Zementis Predictive Analytics 4.1 to 4.2.4.
- Added new ModelManager API to add, remove, update PMML models. Refer to ModelManager API in "API Reference for Predictive Analytics Plug-in EPL (ApamaDoc)".
- Added new ResourceManager API to add or remove resources like custom functions and lookup tables. Refer to ResourceManager API in "API Reference for Predictive Analytics Plug-in EPL (ApamaDoc)".
- Added docker packaging kit for Predictive Analytics Plug-in (this is available only on Linux).
- Samples are moved from `APAMA_HOME/adapters/samples` to `APAMA_HOME/samples/PredictiveAnalytics`.
- Added samples for demonstrating integration with ModelManager API, ResourceManager API, Integration with various kinds of Asset Stores, and building a docker enabled application.
- Added an installer dialog for providing license file for Predictive Analytics Engine. The license file can also be copied to `APAMA_WORK/license` directory after installation.

What's new in 9.10 release

- New service parameters are added to Predictive Analytics Plug-in
  - **subscribeToChannel**. Applications can send input scoring requests to this channel for consumption by the Predictive Analytics plug-in.
  - **sendToChannel**. Predictive analytics plug-in will publish the output predictions to this channel. User applications can subscribe to this channel for receiving responses.
  - **maxBatchSize**. To set the maximum number of input events that will be grouped together and consumed by the plug-in. Default is set to 1000.

2 Predictive Analytics Plug-in

The predictive analytics plug-in is an Apama correlator plug-in for integrating with Zementis Predictive Analytics to score predictive models from within Apama applications. For more information on APIs used by Predictive Analytics Plug-in, see "API Reference for Predictive Analytics Plug-in EPL (ApamaDoc)".

To get started with Predictive Analytics plug-in, see "Working with Predictive Analytics Plug-in" on page 19.

Installing Predictive Analytics Plug-in

To get started with the installation, see *Installing Software AG Products* guide. It is intended for use with the following guides:

- *Using Software AG Installer*. This guide explains how to prepare your machine to use the Software AG Installer, and how to use the Software AG Installer and Software AG Uninstaller to install and uninstall your products.
- *Using Software AG Update Manager*. This guide explains how to use the Software AG Update Manager to install and uninstall fixes on your Software AG products.
- *Upgrading Software AG Products*. This guide contains information on how to upgrade Apama.

The most up-to-date versions of these guides are always available at http://documentation.softwareag.com/ (Empower login required).

When you are installing Predictive Analytics Plug-in using Software AG installer, you are prompted for the Predictive Analytics Engine license file. Optionally, if you specify the license file during the installation process, the license file will be copied to the *SAG_HOME*`/Zementis/adapa-bundle/lib` directory. You can also copy the license file to the *SAG_HOME*`/Zementis/adapa-bundle/lib` directory after the installation process. The license file is mandatory to start the Predictive Analytics Engine.

Note:
The name of the license file must be `zementis.license`.

3 Working with Sample Project

The Predictive Analytics plug-in package ships a sample project **EnergyData** available at `APAMA_HOME/samples/PredictiveAnalytics/EnergyData`. You can use this as a reference for using Predictive Analytics Plug-in.

It is recommended that you copy this sample folder to APAMA_WORK directory rather than running it directly from the installation directory. For Windows users with UAC enabled, this step is required to avoid access denied errors when writing to the sample directory.

Ensure that the Predictive Analytics Engine license file (`zementis.license`) is copied to *`SAG_HOME`*`/Zementis/adapa-bundle/lib` folder.

Running the sample project performs the following tasks:

- Starts correlator, injects the plugin bundle, initialises the Predictive Analytics plug-in by injecting `EnergyDataSample.mon` and the input is sent from `EnergyData.evt`.
- Apama EPL application `EnergyDataSample.mon` performs the following tasks to configure the Predictive Analytics plug-in:
  - Creates a new `ServiceParams` instance and provides details about the PMML model to be loaded.
  - Uses a `ServiceHandlerFactory` to create a new Predictive Analytics plug-in instance with the configured parameters.
  - Receives a callback on `onServiceInitialised` with reference to the newly created `ServiceHandler` after the Predictive Analytics plug-in instance is successfully created.
  - Forwards any requests (`SampleInput`) received to the Predictive Analytics engine by creating a new `com.apama.pa.pmml.Input` request.
  - Consumes responses (`com.apama.pa.pmml.Output`) from the Predictive Analytics engine.

Running the sample project in Software AG Designer

To run the sample project in Software AG Designer, see *Using Apama with Software AG Designer*.

Running the sample project in Apama command prompt

Before you can run any of the Apama tools, you must set up the environment correctly. See "Setting up the environment using the Apama Command Prompt" in *Deploying and Managing Apama Applications*.

To run the sample project in Apama command prompt

1. Change to directory to the location where the sample project is located.
2. Start the correlator with Java support enabled:

```
correlator --java
```

3. Inject the required monitors:

```
engine_inject --java
    "$APAMA_HOME/adapters/lib/Predictive-Analytics-Plugin.jar"
engine_inject --cdp
    "$APAMA_HOME/adapters/monitors/predictive_analytics_plugin_monitors.cdp"
```

4. Inject the MonitorScript file to run the sample:

```
engine_inject "monitors/EnergyDataSample.mon"
```

5. Send the EPL application as input to the sample:

```
engine_send "events/EnergyData.evt"
```

Running the sample project using ant configuration

You can use the ant configuration to run the sample project in Apama command prompt on Windows and UNIX platforms. Before you can run any of the Apama tools, you must set up the environment correctly. See "Setting up the environment using the Apama Command Prompt" in *Deploying and Managing Apama Applications*.

To run the sample project using ant configuration

1. Change to `APAMA_HOME/samples/PredictiveAnalytics/EnergyData` directory.

2. Start the sample project by using the command `ant start`.

   This command performs the following tasks:

   - Starts the correlator with Java support.
   - Injects the plug-in jar file and its associated `predictive_analytics_plugin_monitors.cdp` package to the correlator.
   - Injects the EnergyData sample monitor to the correlator available at `APAMA_HOME/samples/PredictiveAnalytics/EnergyData/monitors/EnergyDataSample.mon`
   - Initializes the plug-in to load the PMML file from `APAMA_HOME/samples/PredictiveAnalytics/EnergyData/model/EnergyDataModel.pmml`
   - Sends sample prediction data.

3. Stop the sample project by using the command `ant stop`.

Sample output

The correlator log displays messages similar to the following:

```
com.apama.pa.pmml.sample.PredictiveAnalytics_EnergyData_Sample
   [1] {"Predicted_Usage":"16.18362364781374"}
```

```
com.apama.pa.pmml.sample.PredictiveAnalytics_EnergyData_Sample
   [1] {"Predicted_Usage":"15.397684338406936"}
com.apama.pa.pmml.sample.PredictiveAnalytics_EnergyData_Sample
   [1] {"Predicted_Usage":"19.12970126490951"}
com.apama.pa.pmml.sample.PredictiveAnalytics_EnergyData_Sample
   [1] {"Predicted_Usage":"15.796460465819097"}
com.apama.pa.pmml.sample.PredictiveAnalytics_EnergyData_Sample
   [1] {"Predicted_Usage":"21.046370444450062"}
com.apama.pa.pmml.sample.PredictiveAnalytics_EnergyData_Sample
   [1] {"PredictiveAnalytics_Error":"Value [NA] is invalid for field [PreUse]."}
```

Errors encountered while scoring requests are sent by populating `PredictiveAnalytics_Error` field in `com.apama.pa.pmml.Output` event.

4 Working with Predictive Analytics Plug-in

To start the plug-in

1. Start the correlator.
2. Inject `Predictive-Analytics-Plugin.jar` located at `APAMA_HOME/adapters/lib` to the correlator.
3. Inject `predictive_analytics_plugin_monitors.cdp` CDP file located at `APAMA_HOME/adapters/monitors` to the correlator.
4. Inject user application EPL.

User application EPL

A user application EPL script should perform the following tasks:

1. Create an instance of `ServiceParams`.

   ```
   com.apama.pa.pmml.ServiceParams serviceParams :=
   com.apama.pa.pmml.ServiceParamsHelper.create();
   ```

2. Set the configuration parameters.

   ```
   serviceParams.setPMMLFileName("PMML_CONFIG_FILE_NAME");
   serviceParams.addResource(CUSTOM_RESOURCE_NAME1);
   serviceParams.addResource(CUSTOM_RESOURCE_NAME2);
   ```

   For a full list of configuration parameters, see "API Reference for Predictive Analytics Plug-in EPL (ApamaDoc)".

3. Request the `ServiceHandlerFactory` to create a new service handler and pass the `ServiceParams`.

   ```
   com.apama.pa.pmml.ServiceHandlerFactory
     .create(com.apama.pa.pmml.ServiceName.Zementis,
     "PREDICTIVE_ANALYTICS_INSTANCE_1",
     serviceParams,
     onServiceInitialised,
     onServiceError);
   ```

   You must pass two additional callbacks to the service handler factory.

   - This callback is called when the PMML file is successfully loaded and the service is initialised. The `ServiceHandler` received in this callback can be used to retrieve the list of models available for this service.

     ```
     action onServiceInitialised(com.apama.pa.pmml.ServiceHandler
     servicehandler) {
       //Implement your application logic here
     }
     ```

   - This callback is called if an error is encountered while loading the PMML file or when there is an issue with the input.

```
action onServiceError(com.apama.pa.pmml.ServiceError serviceError) {
  log "Received Service Error " + serviceError.getErrorMessage() at ERROR;
}
```

4. Create an input event and pass it to the plug-in.

```
com.apama.pa.pmml.Input input := new com.apama.pa.pmml.Input;
input.instanceName := "PREDICTIVE_ANALYTICS_INSTANCE_1";
input.modelName := "SAMPLE_MODEL_NAME";
input.requestId := integer.getUnique().toString();
input.inputFields.add("FIELD_1", "FIELD_1_VALUE");
input.inputFields.add("FIELD_2", "FIELD_2_VALUE");
input.inputFields.add("FIELD_2", "FIELD_3_VALUE");
route input;
```

5. Check for the output event which corresponds to the specified input.

```
com.apama.pa.pmml.Output output;
on all com.apama.pa.pmml.Output
(instanceName="PREDICTIVE_ANALYTICS_INSTANCE_1") : output
{
  log output.toString() at INFO;
  //Do additional processing
}
```

Error handling when processing an input request:

- If there is a significant error while processing the input request, you will receive a callback on the `onServiceError` callback registered during service initialisation.
- Errors and warnings reported by the Predictive Analytics Engine are also propagated through the output event.
  - If any errors are found during scoring, search for `PredictiveAnalytics_Error` in the `outputFields`

    Example:

    ```
    com.apama.pa.pmml.Output("Instance_1","206",
      {"PredictiveAnalytics_Error":"Value [NA] is invalid for field
    [PreUse]."},{})
    ```

  - Warnings reported by the scoring engine are also forwarded in the `outputFields` as `PredictiveAnalytics_Warning_<N>`, where N can be 1, 2, 3 ...

    Example:

    ```
    com.apama.pa.pmml.Output("Instance_1","206",
      {"PredictiveAnalytics_Warning_1":"warning message",
      "Predicted_Usage":"19.980840445004088"},{})
    ```

Managing PMML models

The Predictive Analytics Plug-in supports adding, updating, activating, and removing a PMML model at runtime.

To add, update, activate or remove a model in an EPL script

1. The Predictive Analytics Plug-in also supports managing resources at runtime through ModelManager API. ModelManager API can be accessed by calling `getModelManager()` on the `ServiceHandler` object received in service initialisation `onServiceInitialised` callback.

```
action onServiceInitialised(ServiceHandler servicehandler)
{
  ModelManager modelmanager := serviceHandler.getModelManager();
  //Add Model1 from PMML_PATH1
  modelmanager.addModel("PMML_PATH1");
  //Add Model2 from PMML_PATH2
  modelmanager.addModel("PMML_PATH2");
  ...
  ...
  //Update Model1 from another source PMML_PATH3
  modelmanager.updateModel("Model1","PMML_PATH3");
  ...
  ...

  //remove model when done
  modelmanager.removeModel("Model1");
  modelmanager.removeModel("Model2");
}
```

   Any errors are reported through default callback `onServiceError` of service handler.

   You can add custom callbacks for the above mentioned functions as described below:

```
action onServiceInitialised(ServiceHandler servicehandler)
{
  ModelManager modelmanager := serviceHandler.getModelManager();
  //Add Model1 from PMML_PATH1
  modelmanager.addModelCb("PMML_PATH1", onStatus);
  //Add Model2 from PMML_PATH2
  modelmanager.addModelCb("PMML_PATH2", onStatus);
  ...
  ...
  //Update Model1 from another source PMML_PATH3
  modelmanager.updateModelCb("Model1","PMML_PATH3", onStatus);
  ...
  ...
  //Deactivate Model1
  Model model1 := serviceHandler.getModel("Model1")
  if(model1.isActive()) {
    model1.setActive(false)
  }
  ...
  ...
  //remove model when done
  modelmanager.removeModelCb("Model1", onStatus);
  modelmanager.removeModelCb("Model2", onStatus);
}
action onStatus(ServiceError serviceError)
{
  log "Received status on configured callback:
  "+serviceError.getErrorMessage();
}
```

For more information, see "API Reference for Predictive Analytics Plug-in EPL (ApamaDoc)".

For more information, refer to the samples at `APAMA_HOME/samples/PredictiveAnalytics/EnergyDataModelManager`

Loading custom resources

The Predictive Analytics Plug-in supports adding and removing custom resources like lookup tables and custom functions. You can follow these steps to load and use custom functions in EPL.

To add or remove custom resources in an EPL script

1. Create an instance of `ServiceParams`.

```
com.apama.pa.pmml.ServiceParams serviceParams :=
  (new com.apama.pa.pmml.ServiceParamsHelper).create();
```

2. Add any custom resource either with absolute path or relative path.

```
serviceParams.addResource(CUSTOM_RESOURCE_NAME1);
serviceParams.addResource(CUSTOM_RESOURCE_NAME2);
```

The Predictive Analytics Plug-in also supports adding and removing resources at runtime through ResourceManager API. ResourceManager API can be accessed by calling `getResourceManager()` on the `ServiceHandler` object received in service initialization `onServiceInitialised` callback.

```
action onServiceInitialised(ServiceHandler servicehandler)
{
  serviceHandler := servicehandler;
  ResourceManager resourcemanager := serviceHandler.getResourceManager();
  resourcemanager.addResource("CUSTOM_RESOURCE_NAME1");
  resourcemanager.addResource("CUSTOM_RESOURCE_NAME2");
  resourcemanager.addResource("CUSTOM_RESOURCE_NAME3");

  ModelManager modelmanager := serviceHandler.getModelManager();
  //Add MODEL_NAME1 from PMML_PATH1
  modelmanager.addModel("PMML_PATH1");
  //Add MODEL_NAME2 from PMML_PATH2
  modelmanager.addModel("PMML_PATH2");

  //application code ...
  //remove model when done
  modelmanager.removeModel("MODEL_NAME1");
  modelmanager.removeModel("MODEL_NAME1");
  //remove resource when done
  resourcemanager.removeResource("CUSTOM_RESOURCE_NAME1");
  resourcemanager.removeResource("CUSTOM_RESOURCE_NAME2");
  resourcemanager.removeResource("CUSTOM_RESOURCE_NAME3");

}
```

You can also list the resources that are added by calling `listResources()` function using `ResourceManager` object. The resources are removed automatically when the engine stops, but it is recommended to explicitly remove the unused resources.

For more information, refer to the samples at:

- `APAMA_HOME/samples/PredictiveAnalytics/ECommerceFraud Custom Functions`
- `APAMA_HOME/samples/PredictiveAnalytics/ECommerceFraud Custom Context`

Injecting the Predictive Analytics Plug-in using Ant Macro

You can inject the Predictive Analytics package in to the correlator using the ant macro file.

To inject the Predictive Analytics plug-in in to the correlator

1. Import the ant macro file `${APAMA_HOME}\adapters\ant_macros\predictive-analytics-support-macros.xml` to the ant user script.

2. Add the dependency on 'predictive-analytics-plugin-bundle' ant target to inject the Predictive Analytics Plug-in components in to the correlator.

Connecting Predictive Analytics Plug-in to Messaging Services

Predictive Analytics plug-in can accept and process PMML data received from messaging services like Digital Event Services, MQTT, Universal Messaging and so on.

Working with PMML data received from messaging services

The Predictive Analytics plug-in supports data received from other messaging services through interfaces `addModelFromStream`, `addModelFromStreamCb`, `updateModelFromStream`, `updateModelFromStreamCb` . For more information on interfaces, see ApamaDoc.

```
action onServiceInitialised(ServiceHandler servicehandler)
{
  ModelManager modelmanager := serviceHandler.getModelManager();

  // Add MODEL_NAME of PMML_DATA received as sequence of integers
  // any errors while loading a model will be reported
  // using the default error callback registered during service initialisation

  modelManager.addModelFromStream(MODEL_NAME, PMML_DATA);
  // Alternatively register a new error handler
  //modelManager.addModelFromStreamCb(MODEL_NAME, PMML_DATA, modelConfigError);
  ..
  ..

  // Update an exiting PMML model with another source pmmlData2 received as
  // sequence of integers, any errors while updating the model will be reported
  // using the default error callback registered during service initialisation
```

```
  modelManager.updateModelFromStream(MODEL_NAME, UPDATE_PMML_DATA);
  // Alternatively register a new error handler
  //modelManager.updateModelFromStreamCb
     (MODEL_NAME, UPDATE_PMML_DATA, modelConfigError);
  ...
  ...
}

// Optional error handler for reeporting errors encounted while setting up PMML models
action modelConfigError(ServiceError serviceError)
{
 log "Model Manager Stream Sample, error with Add/Update model" +
   serviceError.getErrorMessage() at ERROR;
}
// Error handler for reeporting errors encounted with the instance
action onServiceError(ServiceError serviceError)
{
  log "Model Manager Stream Sample " + serviceError.getErrorMessage()
   at ERROR;
}
```

Connecting Predictive Analytics Plug-in to Digital Event Services

1. Start Universal Messaging server.
2. Create a digital event definition for sending a PMML model using digital event service.
3. Define a byte array field in the event definition to carry PMML data. You can optionally define additional fields to store information like modelName, add/update/delete model, instanceName and so on.
4. Place the created event in `APAMA_HOME/common/DigitalEventServices/TypeRepository`.

You must configure the Predictive Analytics plug-in to connect to digital event service.

To configure the Predictive Analytics plug-in

1. Create a new Apama project. In the **New Apama Project wizard**, select the **Digital Event Services** bundle and **Predictive Analytics Plug-in** bundle.
2. In the project's **config > connectivity > DigitalEventServices** node, double click **EventTypeList.apamades**.
3. Select the digital event types to convert to Apama event definitions. Edit and save the `EventTypeList.apamades` file.

   Software AG Designer automatically generates the EPL files containing the Apama event definitions for the selected event types in the project's **autogenerated > DigitalEventTypes** node.
4. In the project's **config > connectivity > DigitalEventServices**, double click `DESConnectivity.properties` file and edit the property

`DigitalEventServices_replaceConfigWithRNAME` with realm URL to connect to Universal messaging.

5. In the sample project's monitors, `LoadPmmlFromStreamSample.mon` subscribes to `DES_MODEL_EVENT.CHANNEL` and opens an event listener. Here `addModelFromStream` of `modelManager` is called to process received `pmmlData`.

```
using com.softwareag.connectivity.ConnectivityPlugins;
using DES_MODEL_EVENT;
...
ConnectivityPlugins.onApplicationInitialized();
monitor.subscribe(DES_MODEL_EVENT.CHANNEL);
// This context will now receive digital events of type 'DES_MODEL_EVENT'
action onServiceInitialised(ServiceHandler servicehandler)
{
  ModelManager modelmanager := serviceHandler.getModelManager();
  DES_MODEL_EVENT modelEvt;
  on  all DES_MODEL_EVENT(instanceName=config.instanceName) :modelEvt
  {
    modelManager.addModelFromStream(MODEL_NAME,modelEvt.pmmlData);
  }
}
```

6. Run the Apama project.