

ModelArts

ExeML

Issue 01
Date 2024-03-26



Copyright © Huawei Technologies Co., Ltd. 2024. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: <https://www.huawei.com>

Email: support@huawei.com

Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process*. For details about this process, visit the following web page:

<https://www.huawei.com/en/psirt/vul-response-process>

For vulnerability information, enterprise customers can visit the following web page:

<https://securitybulletin.huawei.com/enterprise/en/security-advisory>

Contents

1 ExeML (New Version)	1
1.1 Introduction to ExeML.....	1
1.2 Image Classification.....	2
1.2.1 Preparing Data.....	3
1.2.2 Creating a Project.....	4
1.2.3 Labeling Data.....	7
1.2.4 Training a Model.....	11
1.2.5 Deploying a Model as a Service.....	12
1.3 Object Detection.....	15
1.3.1 Preparing Data.....	15
1.3.2 Creating a Project.....	18
1.3.3 Labeling Data.....	21
1.3.4 Training a Model.....	25
1.3.5 Deploying a Model as a Service.....	26
1.4 Predictive Analytics.....	30
1.4.1 Preparing Data.....	30
1.4.2 Creating a Project.....	33
1.4.3 Training a Model.....	37
1.4.4 Deploying a Model as a Service.....	38
1.5 Sound Classification.....	40
1.5.1 Preparing Data.....	41
1.5.2 Creating a Project.....	42
1.5.3 Labeling Data.....	45
1.5.4 Training a Model.....	47
1.5.5 Deploying a Model as a Service.....	49
1.6 Text Classification.....	51
1.6.1 Preparing Data.....	51
1.6.2 Creating a Project.....	52
1.6.3 Labeling Data.....	56
1.6.4 Training a Model.....	59
1.6.5 Deploying a Model as a Service.....	60
1.7 Tips.....	63
1.7.1 How Do I Quickly Create an OBS Bucket and a Folder When Creating a Project?.....	63

1.7.2 Where Are Models Generated by ExeML Stored? What Other Operations Are Supported?.....	64
2 ExeML (Old Version).....	66
2.1 Introduction to ExeML.....	66
2.2 Image Classification.....	68
2.2.1 Preparing Data.....	68
2.2.2 Creating a Project.....	69
2.2.3 Labeling Data.....	72
2.2.4 Training a Model.....	75
2.2.5 Deploying a Model as a Service.....	79
2.3 Object Detection.....	81
2.3.1 Preparing Data.....	82
2.3.2 Creating a Project.....	85
2.3.3 Labeling Data.....	87
2.3.4 Training a Model.....	90
2.3.5 Deploying a Model as a Service.....	94
2.4 Predictive Analytics.....	97
2.4.1 Preparing Data.....	97
2.4.2 Creating a Project.....	99
2.4.3 Selecting a Label Column.....	101
2.4.4 Training a Model.....	102
2.4.5 Deploying a Model as a Service.....	103
2.5 Sound Classification.....	105
2.5.1 Preparing Data.....	105
2.5.2 Creating a Project.....	106
2.5.3 Labeling Data.....	109
2.5.4 Training a Model.....	111
2.5.5 Deploying a Model as a Service.....	113
2.6 Text Classification.....	115
2.6.1 Preparing Data.....	116
2.6.2 Creating a Project.....	117
2.6.3 Labeling Data.....	119
2.6.4 Training a Model.....	122
2.6.5 Deploying a Model as a Service.....	125
2.7 Tips.....	127
2.7.1 How Do I Quickly Create an OBS Bucket and a Folder When Creating a Project?.....	128
2.7.2 How Do I View the Added Data in an ExeML Project?.....	128
2.7.3 How Do I Perform Incremental Training in an ExeML Project?.....	129
2.7.4 Where Are Models Generated by ExeML Stored? What Other Operations Are Supported?.....	130
2.7.5 Upgrading a Project Version.....	131

1 ExeML (New Version)

[Introduction to ExeML](#)

[Image Classification](#)

[Object Detection](#)

[Predictive Analytics](#)

[Sound Classification](#)

[Text Classification](#)

[Tips](#)

1.1 Introduction to ExeML

ExeML Functions

ModelArts ExeML is a customized code-free model development tool that helps you start codeless AI application development with high flexibility. ExeML automates model design, parameter tuning and training, and model compression and deployment based on the labeled data. With ExeML, you only need to upload data and perform simple operations as prompted on the ExeML GUI to train and deploy models.

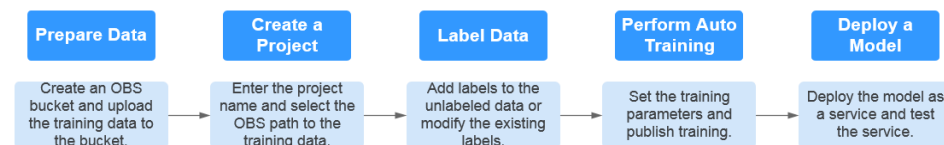
You can use ExeML to quickly build models for sound classification, text classification, image classification, predictive analytics, and object detection. ExeML is widely used in industrial, retail, and security sectors.

- Image classification: identifies a class of objects in images.
- Object detection: identifies the position and class of each object in an image.
- Predictive analytics: classifies or predicts structured data.
- Sound classification: classifies and identifies different sounds.
- Text classification: identifies the category of a piece of text. Currently, only Chinese is supported.

ExeML Process

With ModelArts ExeML, you can develop AI models without coding. You only need to upload data, create a project, label the data, train a model, and deploy the trained model. For details, see [Figure 1-1](#). In the new-version ExeML, this process can be finished by a workflow. You can develop a DAG through a workflow. DAG execution is to use a task execution template to perform data labeling, dataset publishing, model training, model registration, and service deployment in sequence. For more information about workflows, see [Workflow Overview](#).

Figure 1-1 ExeML process



ExeML Projects

- **Image Classification**

An image classification project aims to classify images. You only need to add images and label them. Then, an image classification model can be quickly generated for automatically classifying offerings, vehicle types, and defective goods. For example, in the quality check scenario, you can upload a product image, label the image as qualified or unqualified, and train and deploy a model to inspect product quality.

- **Object Detection**

An object detection project aims to identify the class and location of objects in images. You only need to add images and label objects in the images with proper bounding boxes. The labeled images will be used as a training set for building a model to identify multiple objects or provide the number of objects in a single image. Object detection can also be used to inspect employees' dress code and perform unattended inspection of article placement.

- **Predictive Analytics**

A predictive analytics project is an automated model training application for structured data, which can classify or predict structured data. Predictive analytics can be used for user profile analysis and targeted marketing, as well as predictive maintenance of manufacturing equipment based on real-time data to identify equipment faults.

- **Sound Classification**

A sound classification project identifies whether a certain sound is contained in an audio file. Sound classification can be used to monitor abnormal sounds in production or security scenarios.

- **Text Classification**

A text classification project identifies the class of a piece of text. It can be used in emotion analysis or news classification.

1.2 Image Classification

1.2.1 Preparing Data

Before using ModelArts ExeML to build a model, upload data to an OBS bucket. The OBS bucket and ModelArts must be in the same region.

Requirements on Datasets

- Check that all images are undamaged and in a compatible format. The supported formats are JPG, JPEG, BMP, and PNG.
- Do not store data of different projects in the same dataset.
- Collect at least two classes of images with a similar number of images in each class. Make sure each class has a minimum of 20 images.
- To ensure the prediction accuracy of models, the training samples must be similar to the real-world use cases.
- To ensure the generalization capability of models, datasets should cover all possible scenarios.

Uploading Data to OBS

In this section, the OBS console is used to upload data.

Upload files to OBS according to the following specifications:

- The name of files cannot contain plus signs (+), spaces, or tabs.
- If you do not need to upload training data in advance, create an empty folder to store files generated in the future, for example, **/bucketName/data-cat**.
- If you need to upload images to be labeled in advance, create an empty folder and save the images in the folder. An example of the image directory structure is **/bucketName/data-cat/cat.jpg**.
- If you want to upload labeled images to the OBS bucket, upload them according to the following specifications:
 - The dataset for image classification requires storing labeled objects and their label files (in one-to-one relationship with the labeled objects) in the same directory. For example, if the name of the labeled object is **10.jpg**, the name of the label file must be **10.txt**.

Example of data files:

```
|<dataset-import-path>  
| 10.jpg  
| 10.txt  
| 11.jpg  
| 11.txt  
| 12.jpg  
| 12.txt
```

- Only images in JPG, JPEG, PNG, and BMP formats are supported. When uploading images on the OBS console, ensure that the size of an image does not exceed 5 MB and the total size of images to be uploaded in one attempt does not exceed 8 MB. If the data volume is large, use OBS Browser+ to upload images.
- A label name can contain a maximum of 32 characters, including letters, digits, hyphens (-), and underscores (_).
- The specifications of image classification label files (.txt) are as follows:
Each row contains only one label.


```
flower
book
...
```

Procedure for uploading data to OBS:

Perform the following operations to upload data to OBS for model training and building.

1. Log in to OBS Console and **create a bucket** in the same region as ModelArts. If an available bucket exists, ensure that the OBS bucket and ModelArts are in the same region.
2. **Upload the local data** to the OBS bucket. If you have a large amount of data, use OBS Browser+ to upload data or folders. The uploaded data must meet the dataset requirements of the ExeML project.

NOTE

Upload data from unencrypted buckets. Otherwise, training will fail because data cannot be decrypted.

Creating a Dataset

After the data preparation is completed, create a dataset of the type supported by the project. For details, see [Creating a Dataset](#).

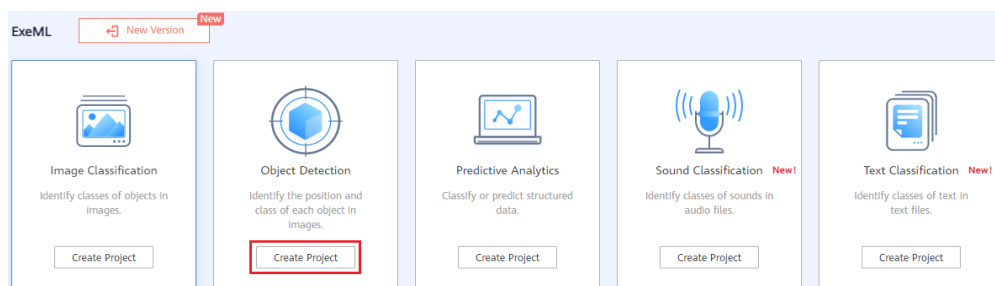
1.2.2 Creating a Project

ModelArts ExeML supports sound classification, text classification, image classification, predictive analytics, and object detection projects. You can create any of them based on your needs. Perform the following operations to create an ExeML project.

Procedure

1. Log in to the ModelArts console. In the navigation pane, choose **ExeML**.
2. Click **Create Project** in the box of your desired project. The page for creating an ExeML project is displayed.

Figure 1-2 Creating a project



3. On the project creation page, set parameters by referring to [Table 1-1](#).

* Billing Mode Pay-per-use

* Name

Description
0/500

* Datasets ↻ Create Dataset

* Output Path Select

* Training Flavor

Table 1-1 Parameters

Parameter	Description
Name	Name of an ExeML project <ul style="list-style-type: none"> Enter a maximum of 64 characters. Only digits, letters, underscores (_), and hyphens (-) are allowed. This parameter is mandatory. Start with a letter. The name must be unique.
Description	Brief description of a project
Dataset	You can select a dataset or click Create Dataset to create one. <ul style="list-style-type: none"> Existing dataset: Select a dataset from the drop-down list box. Only datasets of the same type are displayed. Creating a dataset: Click Create Dataset to create a dataset. For details, see Creating a Dataset.
Output Path	Select an OBS path for storing ExeML data. NOTE The output path stores all data generated in the ExeML project.
Training Flavor	Select a training flavor for this ExeML project. You will be billed based on different flavors.

- Click **Create Project**. Then, the ExeML workflow is displayed.
- Wait until the workflow of the image classification project executes the following phases in sequence:

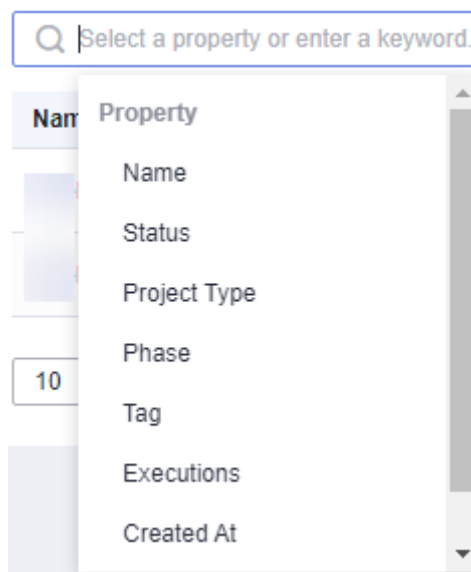
- a. **Label Data:** Check data labeling.
- b. **Publish Dataset Version:** Publish a version for the labeled dataset.
- c. **Check Data:** Check whether any exception occurs in your dataset.
- d. **Classify Images:** Train the dataset of the published version to generate a model.
- e. **Register Model:** Register the trained model with model management.
- f. **Deploy Service:** Deploy the generated model as a real-time service.

Quickly Searching for a Project

On the ExeML overview page, you can use the search box to quickly search for and filter workflows based on the ExeML type (or project name).

1. Log in to the ModelArts console. In the navigation pane, choose **ExeML**.
2. In the search box above the ExeML project list, filter the desired workflows based on the required property, such as name, status, project type, current phase, and tag.

Figure 1-3 Property




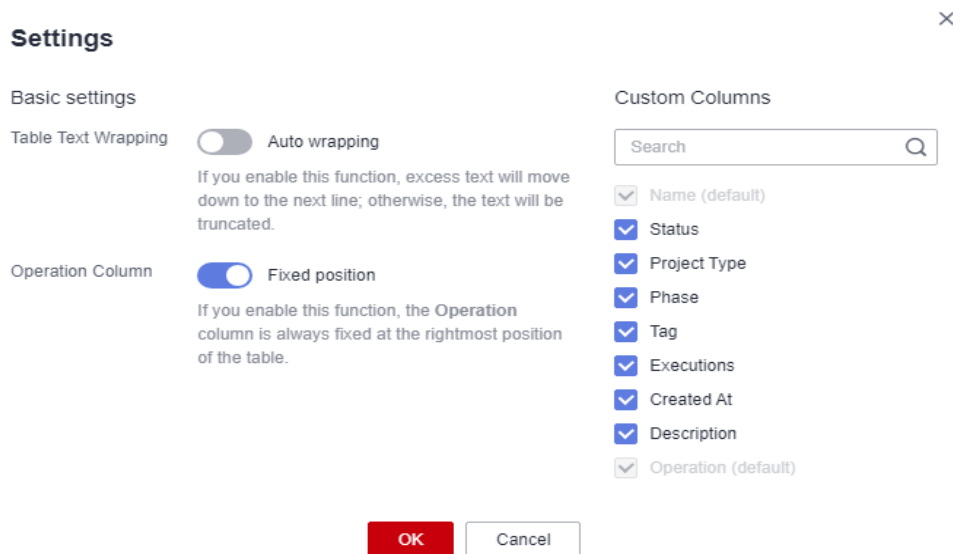

3. To adjust the basic settings of ExeML and select the columns you want to see, click  on the right of the search box.

Table Text Wrapping: This function is disabled by default. If you enable this function, excess text will move down to the next line; otherwise, the text will be truncated.

Operation Column: This function is enabled by default. If you enable this function, the **Operation** column is always fixed at the rightmost position of the table.

Custom Columns: By default, all items are selected. You can select columns you want to see.

Figure 1-4 Customizing table columns

4. Click **OK**. Then, the columns will be displayed based on the settings.
5. To arrange ExeML projects by a specific property, click  in the table header.

1.2.3 Labeling Data

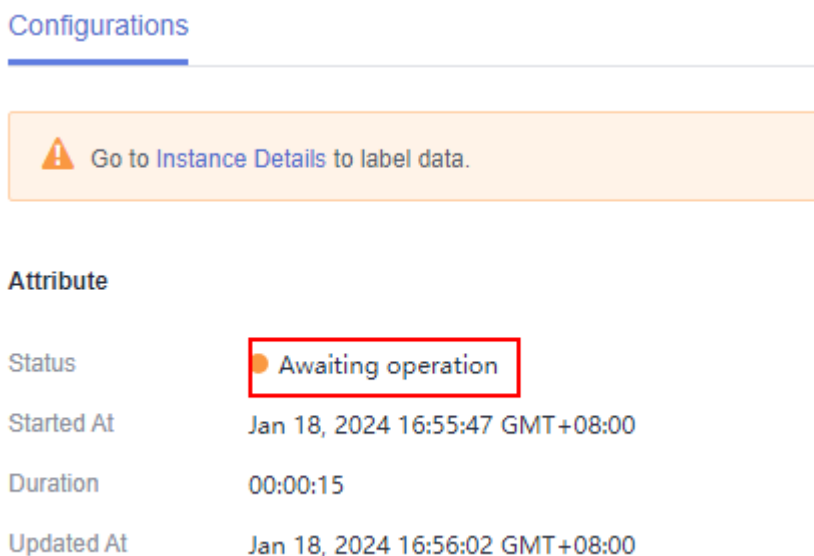
Model training requires a large number of labeled images. Therefore, before model training, add labels to the images that are not labeled. ModelArts allows you to add labels in batches by one click. You can also modify or delete labels that have been added to images.

NOTE

The number of labeled images in the dataset must be no fewer than 100. Otherwise, checking the dataset will fail, affecting your model training.

After the project is created, you will be directed to the ExeML page and the project starts to run. Click the data labeling phase. After the status changes to **Awaiting operation**, confirm the data labeling status in the dataset. You can also modify labels, add data, or delete data in the dataset.

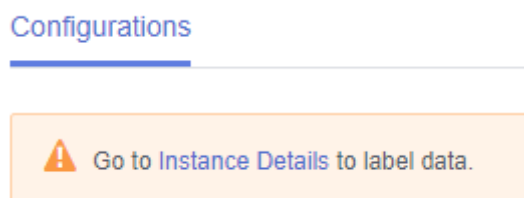
Figure 1-5 Data labeling status



Labeling Images

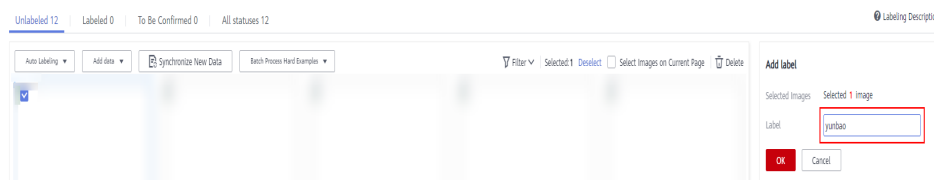
1. On the labeling phase of the new-version ExeML, click **Instance Details**. The data labeling page is displayed.

Figure 1-6 Clicking Instance Details



2. Select the images to be labeled in sequence, or tick **Select Images on Current Page** to select all images on the page, and then add labels to the images in the right pane.

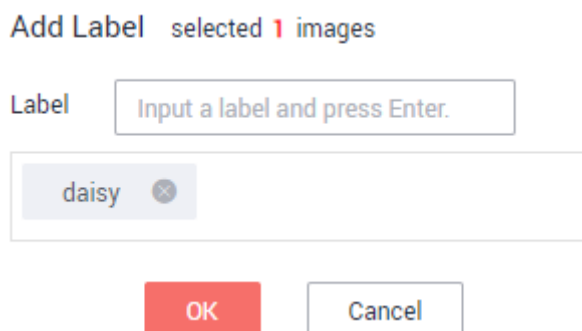
Figure 1-7 Labeling an image



3. After selecting an image, input a label in the **Label** text box, or select an existing label from the drop-down list. Click **OK**. The selected image is labeled. For example, you can select multiple images containing tulips and add label **tulips** to them. Then select other unlabeled images and label them as **sunflowers** and **roses**. After the labeling is complete, the images are saved on the **Labeled** tab page.
 - a. You can add multiple labels to an image.

- b. A label consists of letters, digits, hyphens (-), and underscores (_).

Figure 1-8 Image labeling



- 4. After all the images are labeled, view them on the **Labeled** tab page or view **All Labels** in the right pane to check the name and quantity of the labels.

Synchronizing or Adding Images

On the labeling phase, click **Instance Details** to go to the data labeling page. Then, add images from your local PC or synchronize image data from OBS.

Figure 1-9 Adding local images

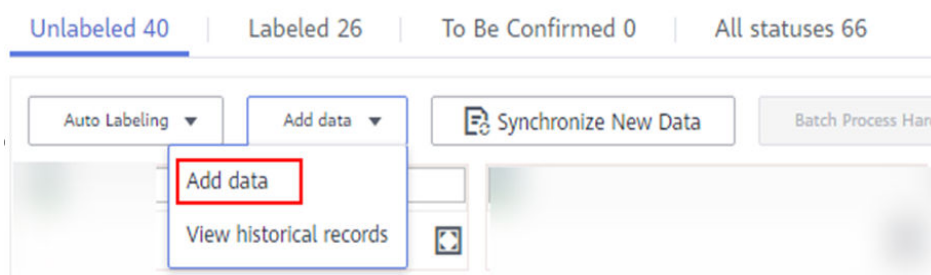
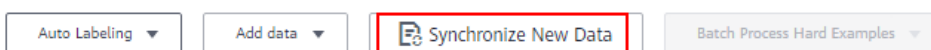


Figure 1-10 Synchronizing OBS images



- **Add data:** You can click **Add data** to quickly add images on a local PC to ModelArts. These images will be automatically synchronized to the OBS path specified during project creation.
- **Synchronize New Data:** You can upload images to the OBS directory specified during project creation and click **Synchronize New Data** to quickly add the images in the OBS directory to ModelArts.
- **Delete Image:** You can delete images one by one, or tick **Select Current Page** to delete all images on the page.

NOTE

The deleted images cannot be recovered. Exercise caution when performing this operation.

Modifying Labeled Data

After labeling data, you can modify the labeled data on the **Labeled** tab page.

- **Modifying based on images**

On the data labeling page, click the **Labeled** tab, and select one or more images to be modified from the image list. Modify the image information in the label information area on the right.


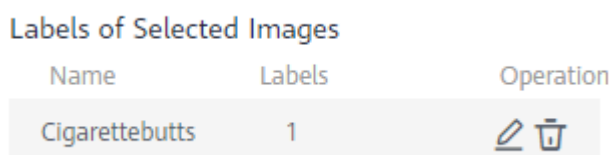



- Adding a label: In the **Label** text box, select an existing label or enter a new label name, and then click .
- Modifying a label: In the **Labels of Selected Images** area, click the editing icon in the **Operation** column, enter the correct label name in the text box, and click the check mark icon to complete the modification.

Figure 1-11 Modifying a label



Name	Labels	Operation
Cigarettebutts	1	 

- Deleting a label: In the **Labels of Selected Images** area, click  in the **Operation** column to delete the label.

- **Modifying based on labels**

On the labeling overview page, click **Label Management**. Information about all labels is displayed.

Figure 1-12 Information about all labels



Label Name	Attribute	Label Color	Operation
<input type="checkbox"/> YUNBAO	--		Modify Delete

- Modifying a label: In the **Operation** column of the target label, click **Modify**, enter the new label, and click **OK**.
- Deleting a label: In the **Operation** column of the target label, click **Delete**, and click **OK**.

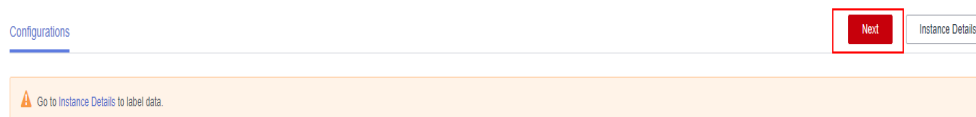
 **NOTE**

Deleted tags cannot be restored.

Resuming Workflow Execution

After confirming data labeling, return back to the new-version ExeML. Click **Next**. Then, the workflow continues to run in sequence until all phases are executed.

Figure 1-13 Resuming the workflow execution



1.2.4 Training a Model

After labeling the images, perform model training to obtain the required image classification model. Ensure that the labeled images meet the requirements specified in [Prerequisites](#). Otherwise, checking the dataset will fail.

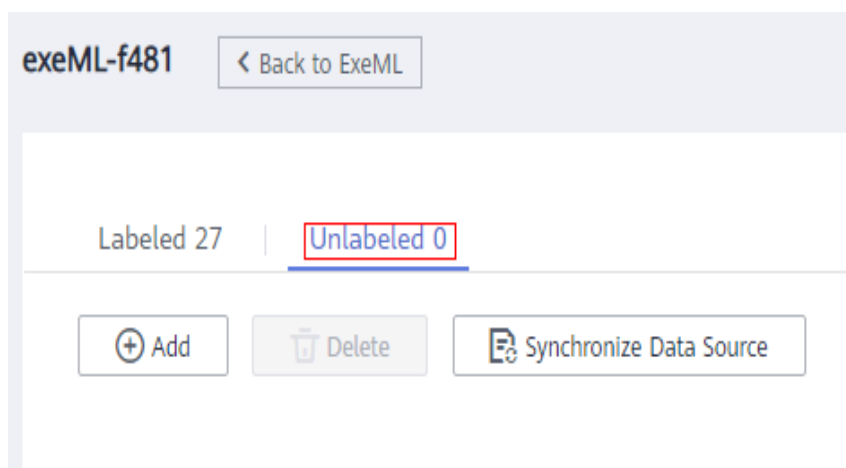
Prerequisites

1. The number of labeled images in your dataset is greater than or equal to 100.
2. At least two classes of samples are required for training, and each class with at least 5 samples.

Procedure

1. Ensure all your dataset has been labeled. For details, see [Labeling Data](#).

Figure 1-14 Finding unlabeled data




2. In the data labeling phase of the new-version ExeML, click **Next** and wait until the workflow enters the training phase.
3. Wait until the training is complete. No manual operation is required. If you close or exit the page, the system continues training until it is complete.
4. On the image classification phase, wait until the training status changes from **Running** to **Completed**.
5. After the training, click  on the image classification phase to view metric information. For details about the evaluation result parameters, see [Table 1-2](#).

Table 1-2 Evaluation result parameters

Parameter	Description	Description
Recall	Recall	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish positive samples.

Parameter	Description	Description
Precision	Precision	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish negative samples.
Accuracy	Accuracy	Fraction of correctly predicted samples over all samples. It shows the general ability of a model to recognize samples.
F1 Score	F1 score	Harmonic average of the precision and recall of a model. It is used to evaluate the quality of a model. A high F1 score indicates a good model.

 NOTE

An ExeML project supports multiple rounds of training, and each round generates an AI application version. For example, the first training version is **0.0.1**, and the next version is **0.0.2**. The trained models can be managed by training version. After the trained model meets your requirements, deploy the model as a service.

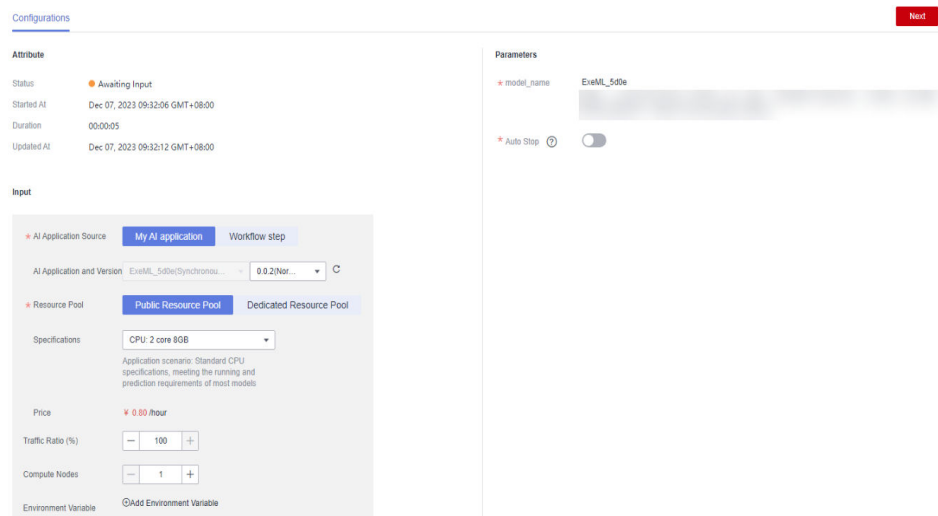
1.2.5 Deploying a Model as a Service

Deploying a Service

You can deploy a model as a real-time service that provides a real-time test UI and monitoring capabilities. After model training is complete, you can deploy a version with the ideal accuracy and in the **Successful** status as a service. The procedure is as follows:

1. On the phase execution page, after the service deployment status changes to **Awaiting input**, double-click **Deploy Service**. On the configuration details page, configure resource parameters.
2. On the service deployment page, select the resource specifications used for service deployment.

Figure 1-15 Resource specifications



- **AI Application Source:** defaults to the generated AI application.
- **AI Application and Version:** The current AI application version is automatically selected, which is changeable.
- **Resource Pool:** defaults to public resource pools.
- **Traffic Ratio:** defaults to **100** and supports a value range of 0 to 100.
- **Specifications:** Select available specifications based on the list displayed on the console. The specifications in gray cannot be used in the current environment. If there are no specifications after you select a public resource pool, no public resource pool is available in the current environment. In this case, use a dedicated resource pool or contact the administrator to create a public resource pool.
- **Compute Nodes:** an integer ranging from 1 to 5. The default value is **1**.
- **Auto Stop:** enables a service to automatically stop at a specified time. If this function is not enabled, the real-time service continuously runs and fees are incurred accordingly. Auto stop is enabled by default and its default value is **1 hour later**.

The auto stop options are **1 hour later**, **2 hours later**, **4 hours later**, **6 hours later**, and **Custom**. If you select **Custom**, enter any integer from 1 to 24 in the text box on the right.

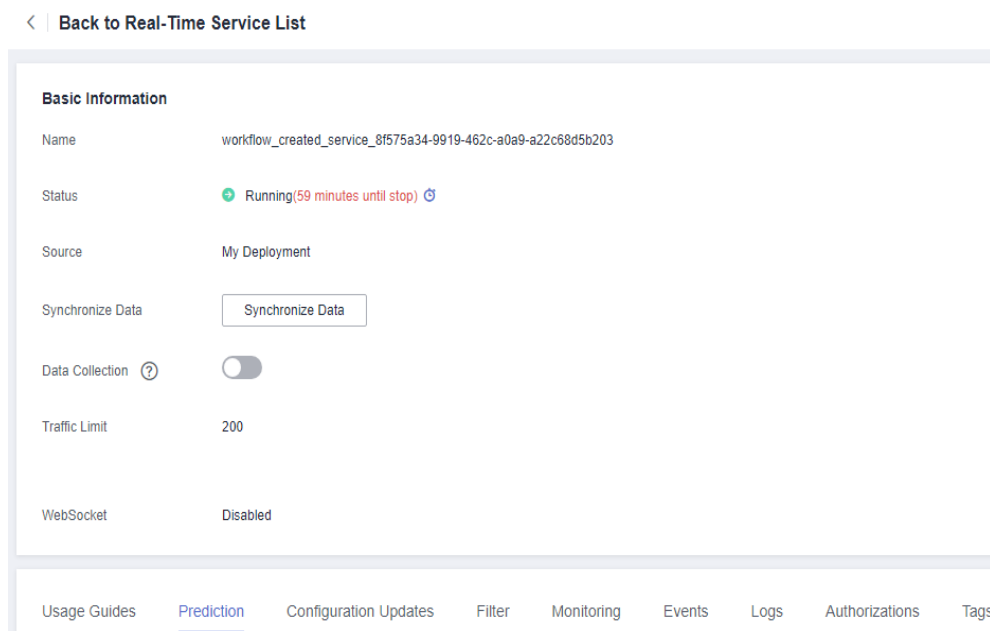
NOTE

You can choose the package that you have bought when you select specifications. On the configuration fee tag, you can view your remaining package quota and how much you will pay for any extra usage.

3. After configuring resources, click **Next**. Wait until the status changes to **Executed**. The AI application has been deployed as a real-time service.

Testing a Service

- After the service is deployed, click **Instance Details** to go to the real-time service details page. Click the **Prediction** tab to test the service.

Figure 1-16 Testing the service

- You can also choose **Service Deployment** > **Real-Time Services** and click **Predict** in the **Operation** column of the target service for testing. The testing procedure is the same as that described in the following section. For details, see [Testing the Deployed Service](#).
- You can also use code to test a service. For details, see [Accessing Real-Time Services](#).
- The following describes the procedure for performing a service test after the image classification model is deployed as a service on the ExeML page.
 - a. After the model is deployed, click **Instance Details** in the service deployment phase to go to the service page. On the **Prediction** tab page, click **Upload** and select a local image for test.
 - b. Click **Prediction** to conduct the test. After the prediction is complete, label **sunflowers** and its detection score are displayed in the prediction result area on the right. If the model accuracy does not meet your expectation, add images on the **Label Data** tab page, label the images, and train and deploy the model again. [Table 1-3](#) describes the parameters in the prediction result. If you are satisfied with the model prediction result, call the API to access the real-time service as prompted. For details, see [Accessing Real-Time Services](#).
Only JPG, JPEG, BMP, and PNG images are supported.

Figure 1-17 Prediction result



```

1 {
2   "predicted_label": "sunflowers",
3   "scores": [
4     [
5       "sunflowers",
6       "0.972"
7     ],
8     [
9       "dandelion",
10      "0.028"
11     ],
12     [
13       "daisy",
14       "0.000"
15     ],
16     [
17       "roses",
18       "0.000"
19     ],
20     [
21       "tulips",
22       "0.000"
23     ]
24   ]
25 }

```

Table 1-3 Parameters in the prediction result

Parameter	Description
predicted_label	Image prediction label
scores	Prediction confidence of top 5 labels

NOTE

- A running real-time service continuously consumes resources. If you do not need to use the real-time service, stop the service to stop billing. To do so, click **Stop** in the **More** drop-down list in the **Operation** column. If you want to use the service again, click **Start**.
- If you enable the auto stop function, the service automatically stops after the specified time and no fee is generated.

1.3 Object Detection

1.3.1 Preparing Data

Before using ModelArts ExeML to build a model, upload data to an OBS bucket. The OBS bucket and ModelArts must be in the same region.

Uploading Data to OBS

This operation uses the OBS console to upload data.

Perform the following operations to import data to the dataset for model training and building.

1. Log in to OBS Console and **create a bucket** in the same region as ModelArts. If an available bucket exists, ensure that the OBS bucket and ModelArts are in the same region.
2. **Upload the local data** to the OBS bucket. If you have a large amount of data, use OBS Browser+ to upload data or folders. The uploaded data must meet the dataset requirements of the ExeML project.

 NOTE

Upload data from unencrypted buckets. Otherwise, training will fail because data cannot be decrypted.

Requirements on Datasets

- The name of files in a dataset cannot contain Chinese characters, plus signs (+), spaces, or tabs.
- Ensure that no damaged image exists. The supported image formats include JPG, JPEG, BMP, and PNG.
- Do not store data of different projects in the same dataset.
- To ensure the prediction accuracy of models, the training samples must be similar to the actual application scenarios.
- To ensure the generalization capability of models, datasets should cover all possible scenarios.
- In an object detection dataset, if the coordinates of the bounding box exceed the boundaries of an image, the image cannot be identified as a labeled image.

Requirements for Files Uploaded to OBS

- If you do not need to upload training data in advance, create an empty folder to store files generated in the future, for example, **/bucketName/data-cat**.
- If you need to upload images to be labeled in advance, create an empty folder and save the images in the folder. An example of the image directory structure is **/bucketName/data-cat/cat.jpg**.
- If you want to upload labeled images to the OBS bucket, upload them according to the following specifications:
 - The dataset for object detection requires storing labeled objects and their label files (in one-to-one relationship with the labeled objects) in the same directory. For example, if the name of the labeled object is **IMG_20180919_114745.jpg**, the name of the label file must be **IMG_20180919_114745.xml**.

The label files for object detection must be in PASCAL VOC format. For details about the format, see [Table 1-4](#).

Example of data files:

```
├─<dataset-import-path>  
  IMG_20180919_114732.jpg  
  IMG_20180919_114732.xml  
  IMG_20180919_114745.jpg  
  IMG_20180919_114745.xml  
  IMG_20180919_114945.jpg  
  IMG_20180919_114945.xml
```

- Images in JPG, JPEG, PNG, and BMP formats are supported. When uploading images on the ModelArts console, ensure that the size of an image does not exceed 5 MB and the total size of images to be uploaded in one attempt does not exceed 8 MB. If the data volume is large, use OBS Browser+ to upload images.
- A label name can contain a maximum of 32 characters, including letters, digits, hyphens (-), and underscores (_).

Table 1-4 PASCAL VOC format description

Field	Mandatory	Description
folder	Yes	Directory where the data source is located
filename	Yes	Name of the file to be labeled
size	Yes	Image pixel <ul style="list-style-type: none">• width: image width. This parameter is mandatory.• height: image height. This parameter is mandatory.• depth: number of image channels. This parameter is mandatory.
segmented	Yes	Segmented or not
object	Yes	Object detection information. Multiple object{} functions are generated for multiple objects. <ul style="list-style-type: none">• name: class of the labeled object. This parameter is mandatory.• pose: shooting angle of the labeled object. This parameter is mandatory.• truncated: whether the labeled object is truncated (0 indicates that the object is not truncated). This parameter is mandatory.• occluded: whether the labeled object is occluded (0 indicates that the object is not occluded). This parameter is mandatory.• difficult: whether the labeled object is difficult to identify (0 indicates that the object is easy to identify). This parameter is mandatory.• confidence: confidence score of the labeled object. The value range is 0 to 1. This parameter is optional.• bndbox: bounding box type. This parameter is mandatory. For details about the possible values, see Table 1-5.

Table 1-5 Description of bounding box types

type	Shape	Labeling Information
bndbox	Rectangle	Coordinates of the upper left and lower right points <xmin>100<xmin> <ymin>100<ymin> <xmax>200<xmax> <ymin>200<ymin>

Example of the label file in KITTI format:

```
<annotation>
  <folder>test_data</folder>
  <filename>260730932.jpg</filename>
  <size>
    <width>767</width>
    <height>959</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>bag</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <occluded>0</occluded>
    <difficult>0</difficult>
    <bndbox>
      <xmin>108</xmin>
      <ymin>101</ymin>
      <xmax>251</xmax>
      <ymin>238</ymin>
    </bndbox>
  </object>
</annotation>
```

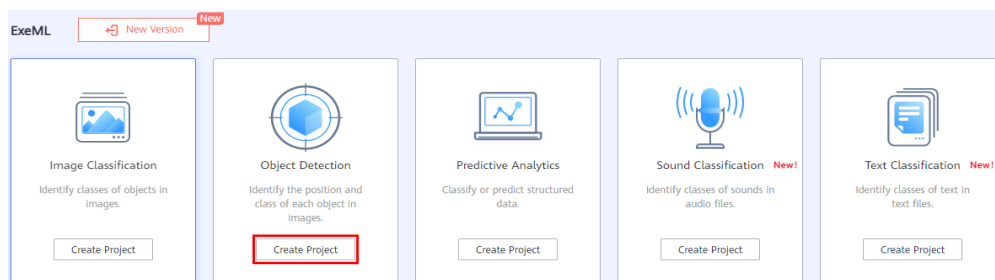
1.3.2 Creating a Project

ModelArts ExeML supports sound classification, text classification, image classification, predictive analytics, and object detection projects. You can create any of them based on your needs. Perform the following operations to create an ExeML project.

Procedure

1. Log in to the ModelArts console. In the navigation pane, choose **ExeML**.
2. Click **Create Project** in the box of your desired project. The page for creating an ExeML project is displayed.

Figure 1-18 Create Project



3. On the project creation page, set parameters by referring to [Table 1-6](#).

* Billing Mode Pay-per-use

* Name

Description 0/500

* Datasets ↻ Create Dataset

* Output Path Select

* Training Flavor

Table 1-6 Parameters

Parameter	Description
Name	Name of a project <ul style="list-style-type: none"> Enter a maximum of 64 characters. Only digits, letters, underscores (_), and hyphens (-) are allowed. This parameter is mandatory. Start with a letter. The name must be unique.
Description	Brief description of a project
Dataset Source	You can select a dataset or click Create Dataset to create one. <ul style="list-style-type: none"> Existing dataset: Select a dataset from the drop-down list box. Only datasets of the same type are displayed. Creating a dataset: Click Create Dataset to create a dataset. For details, see Creating a Dataset.

Parameter	Description
Output Path	An OBS path for storing ExeML data NOTE The output path stores all data generated in the ExeML project.
Training Flavor	Select a training flavor for this ExeML project. You will be billed based on different flavors.

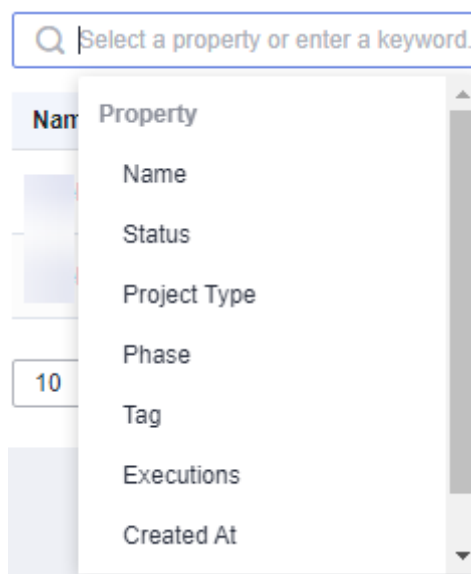
4. Click **Create Project**. Then, the ExeML workflow is displayed.
5. Wait until the workflow of the object detection project executes the following phases in sequence:
 - a. **Label Data**: Check data labeling.
 - b. **Publish Dataset Version**: Publish a version for the labeled dataset.
 - c. **Check Data**: Check whether any exception occurs in your dataset.
 - d. **Detect Objects**: Train the dataset of the published version to generate a model.
 - e. **Register Model**: Register the trained model with model management.
 - f. **Deploy Service**: Deploy the generated model as a real-time service.

Quickly Searching for a Project

On the ExeML overview page, you can use the search box to quickly search for and filter workflows based on the ExeML type (or project name).

1. Log in to the ModelArts console. In the navigation pane, choose **ExeML**.
2. In the search box above the ExeML project list, filter the desired workflows based on the required property, such as name, status, project type, current phase, and tag.

Figure 1-19 Property




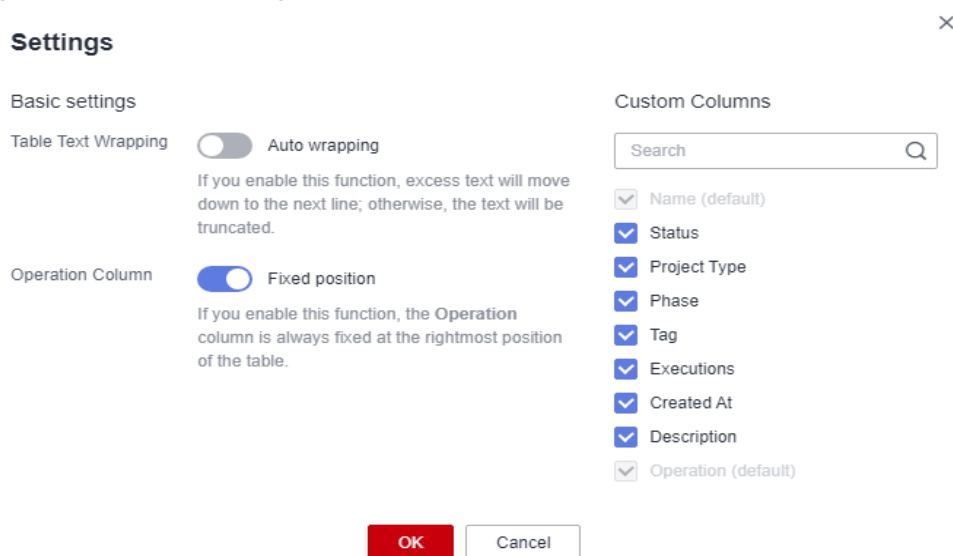
- To adjust the basic settings of ExeML and select the columns you want to see, click  on the right of the search box.


Table Text Wrapping: This function is disabled by default. If you enable this function, excess text will move down to the next line; otherwise, the text will be truncated.

Operation Column: This function is enabled by default. If you enable this function, the **Operation** column is always fixed at the rightmost position of the table.

Custom Columns: By default, all items are selected. You can select columns you want to see.

Figure 1-20 Customizing table columns



- Click **OK**. Then, the columns will be displayed based on the settings.
- To arrange ExeML projects by a specific property, click  in the table header.

1.3.3 Labeling Data

Before data labeling, consider how to design labels. The labels must correspond to the distinct characteristics of the detected images and are easy to identify (the detected object in an image is highly distinguished from the background). Each label specifies the expected recognition result of the detected images. After the label design is complete, prepare images based on the designed labels. It is recommended that the number of all images to be detected be greater than 100. If the labels of some images are similar, prepare more images. At least two classes of samples are required for training, and each class with at least 50 samples.

- During labeling, the variance of a class should be as small as possible. That is, the labeled objects of the same class should be as similar as possible. The labeled objects of different classes should be as different as possible.
- The contrast between the labeled objects and the image background should be as stark as possible.

- In object detection labeling, a target object must be entirely contained within a labeling box. If there are multiple objects in an image, do not relabel or miss any objects.


After a project is created, you will be redirected to the new-version ExeML and the project starts to run. When the data labeling phase changes to **Awaiting operation**, manually confirm data labeling in the dataset. You can also add or delete data in the dataset and modify labels.

Figure 1-21 Data labeling status

Configurations

 Go to [Instance Details](#) to label data.

Attribute

Status	 Awaiting operation
Started At	Jan 18, 2024 16:55:47 GMT+08:00
Duration	00:00:15
Updated At	Jan 18, 2024 16:56:02 GMT+08:00

Labeling Images

1. On the labeling phase of the new-version ExeML, click **Instance Details**. The data labeling page is displayed. Click an image to go to the labeling page.

Configurations

 Go to [Instance Details](#) to label data.

2. Left-click and drag the mouse to select the area where the target object is located. In the dialog box that is displayed, select the label color, enter the label name, for example, **yunbao**, and press **Enter**. After the labeling is complete, the status of the images changes to **Labeled**.

More descriptions of data labeling are as follows:

- You can click the arrow keys in the upper and lower parts of the image, or press the left and right arrow keys on the keyboard to select another image. Then, repeat the preceding operations to label the image. If an image contains more than one object, you can label all the objects.
- You can add multiple labels with different colors for an object detection ExeML project for easy identification. After selecting an object, select a new color and enter a new label name in the dialog box that is displayed to add a new label.

- In an ExeML project, object detection supports only rectangular labeling boxes. In the **Data Management** function, more types of labeling boxes are supported for object detection datasets.
- In the **Label Data** window, you can scroll the mouse to zoom in or zoom out on the image to quickly locate the object.

Figure 1-22 Image labeling for object detection



NOTE

For an object detection dataset, you can add multiple labeling boxes and labels to an image during labeling. The labeling boxes cannot extend beyond the image boundary.

3. After all images in the image directory are labeled, return to the ExeML workflow page and click **Next**. The workflow automatically publishes a data labeling version and performs training.

Synchronizing or Adding Images

In the labeling phase, click **Instance Details** to go to the data labeling page. Then, add images from your local PC or synchronize images from OBS.

Figure 1-23 Adding local images

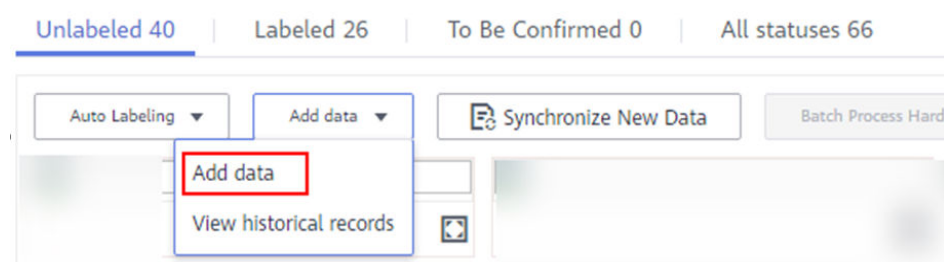
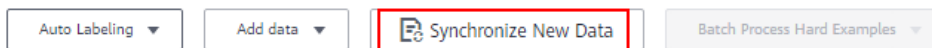


Figure 1-24 Synchronizing images from OBS



- **Add data:** You can quickly add images on a local PC to ModelArts. These images will be automatically synchronized to the OBS path specified during project creation. Click **Add data** and import data.
- **Synchronize New Data:** You can upload images to the OBS directory specified during project creation and click **Synchronize New Data** to quickly add the new images in the OBS directory to ModelArts.
- **Delete:** You can delete images one by one, or select **Select Images on Current Page** to delete all images on the page.

NOTE

Deleted images cannot be recovered.

Modifying Labeled Data

After labeling data, you can modify the labeled data on the **Labeled** tab page.

- **Modifying based on images**

On the dataset details page, click the **Labeled** tab, and then select the image to be modified. Modify the image information in the label information area on the right.

- **Modifying a label:** In the **Labeling** area, click the editing icon, enter the correct label name in the text box, and click the check mark to complete the modification. The label color cannot be modified.
- **Deleting a label:** In the **Labeling** area, click the deletion button to delete a label for the image.

After the label is deleted, click the project name in the upper left corner of the page to exit the labeling page. The image will be returned to the **Unlabeled** tab page.

Figure 1-25 Editing an object detection label

Labels of Selected Images

Label	Count	Operation
sunflower	1	✓ ✕
daisy	2	✎ 🗑️

- **Modifying based on labels**

On the labeling job overview page, click **Label Management** on the right. You will see the label management page, which shows information about all labels.

Figure 1-26 Label management page

Label Name	Attribute	Label Color	Operation
YUNBAO	--	■	Modify Delete

- Modifying a label: Click **Modify** in the **Operation** column. In the dialog box that appears, enter a new label and click **OK**. After the modification, the images that have been added with the label use the new label name.
- Deleting a label: Click the deletion button in the **Operation** column. In the dialog box that appears, confirm the operation and click **OK**.

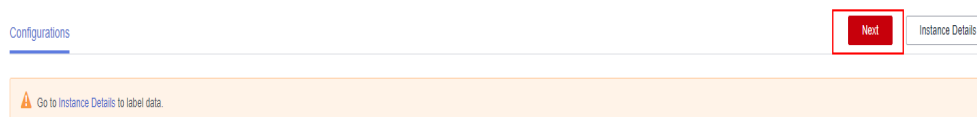
 **NOTE**

Deleted tags cannot be restored.

Resuming Workflow Execution

After confirming data labeling, return back to the new-version ExeML. Click **Next**. Then, the workflow continues to run in sequence until all phases are executed.

Figure 1-27 Resuming the workflow execution



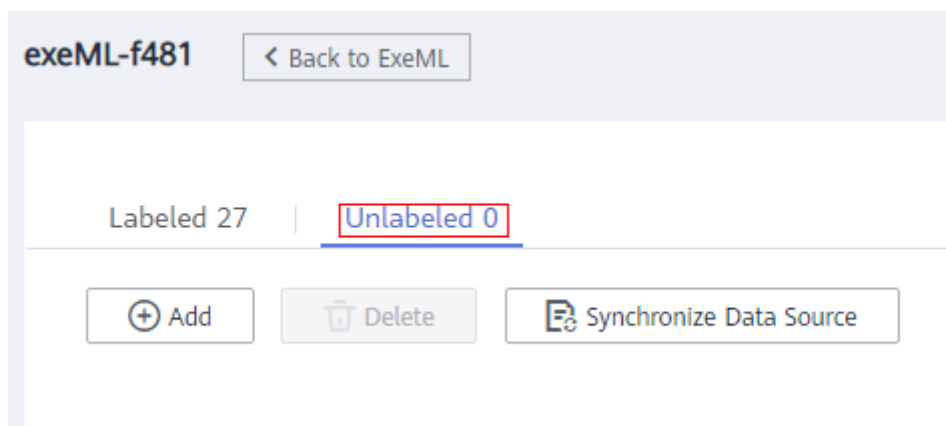
1.3.4 Training a Model

After labeling the images, perform auto training to obtain an appropriate model version.

Procedure

1. On the ExeML page of the new version, click the name of the target project. Then, click **Instance Details** on the labeling phase to label data.

Figure 1-28 Finding unlabeled data



2. Return to the labeling phase of the new-version ExeML, click **Next** and wait until the workflow enters the training phase.
3. Wait until the training is complete. No manual operation is required. If you close or exit the page, the system continues training until it is complete.
4. On the object detection phase, wait until the training status changes from **Running** to **Completed**.


5. After the training, click  on the object detection phase to view metric information. For details about the evaluation result parameters, see [Table 1-7](#).

Table 1-7 Evaluation result parameters

Parameter	Description
Recall	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish positive samples.
Precision	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish negative samples.
Accuracy	Fraction of correctly predicted samples over all samples. It shows the general ability of a model to recognize samples.
F1 Score	Harmonic average of the precision and recall of a model. It is used to evaluate the quality of a model. A high F1 score indicates a good model.

 NOTE

An ExeML project supports multiple rounds of training, and each round generates an AI application version. For example, the first training version is **0.0.1**, and the next version is **0.0.2**. The trained models can be managed by training version. After the trained model meets your requirements, deploy the model as a service.

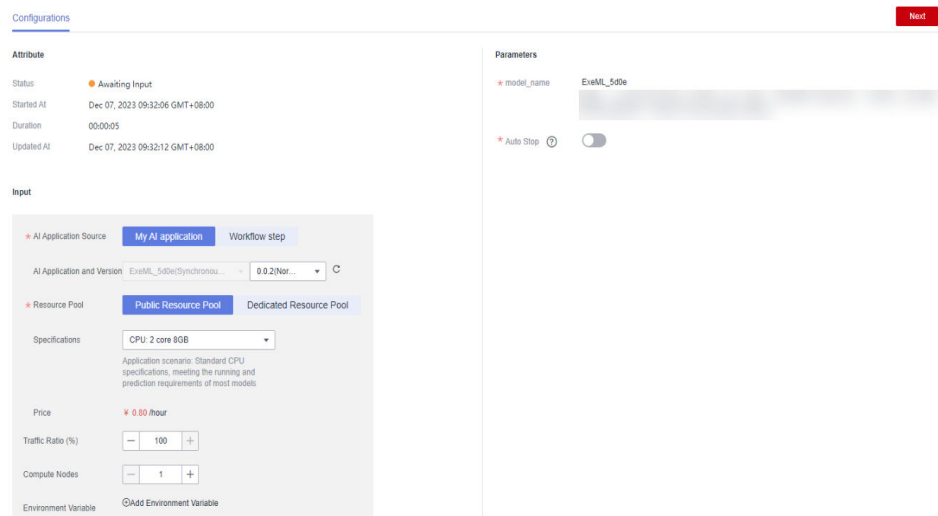
1.3.5 Deploying a Model as a Service

Deploying a Service

You can deploy a model as a real-time service that provides a real-time test UI and monitoring capabilities. After the model is trained, you can deploy a **Successful** version with ideal accuracy as a service. The procedure is as follows:

1. On the phase execution page, after the service deployment status changes to **Awaiting input**, double-click **Deploy Service**. On the configuration details page, configure resource parameters.
2. On the service deployment page, select the resource specifications used for service deployment.

Figure 1-29 Resource specifications



- **AI Application Source:** defaults to the generated AI application.
- **AI Application and Version:** The current AI application version is automatically selected, which is changeable.
- **Resource Pool:** defaults to public resource pools.
- **Traffic Ratio:** defaults to **100** and supports a value range of 0 to 100.
- **Specifications:** Select available specifications based on the list displayed on the console. The specifications in gray cannot be used in the current environment. If there are no specifications after you select a public resource pool, no public resource pool is available in the current environment. In this case, use a dedicated resource pool or contact the administrator to create a public resource pool.
- **Compute Nodes:** an integer ranging from 1 to 5. The default value is **1**.
- **Auto Stop:** enables a service to automatically stop at a specified time. If this function is not enabled, the real-time service continuously runs and fees are incurred accordingly. Auto stop is enabled by default and its default value is **1 hour later**.

The auto stop options are **1 hour later**, **2 hours later**, **4 hours later**, **6 hours later**, and **Custom**. If you select **Custom**, enter any integer from 1 to 24 in the text box on the right.

NOTE

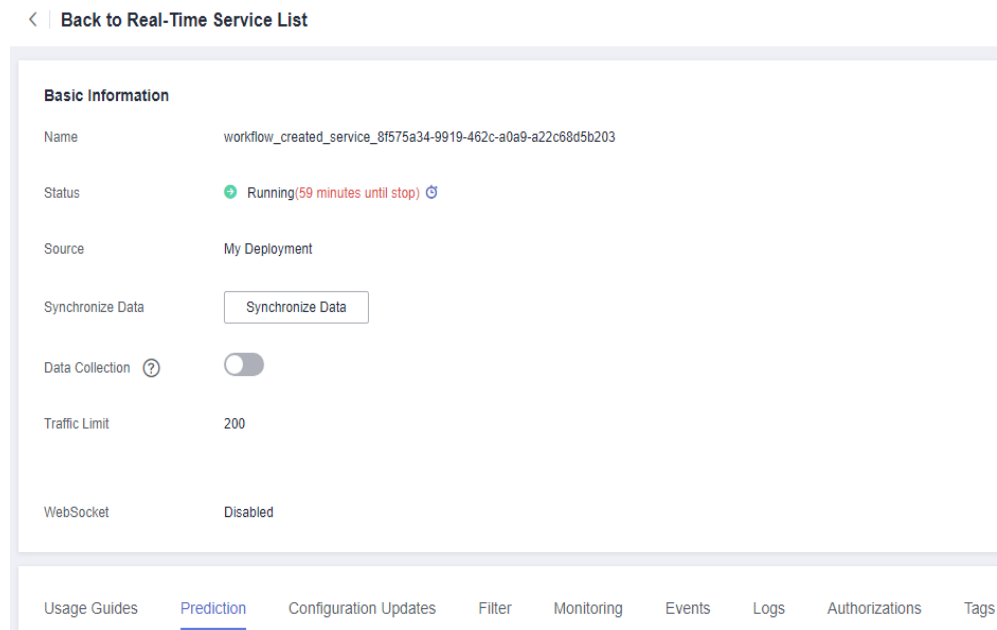
You can choose the package that you have bought when you select specifications. On the configuration fee tag, you can view your remaining package quota and how much you will pay for any extra usage.

3. After configuring resources, click **Next**. Wait until the status changes to **Executed**. The AI application has been deployed as a real-time service.

Testing a Service

- After the service is deployed, click **Instance Details** to go to the real-time service details page. Click the **Prediction** tab to test the service.

Figure 1-30 Testing the service



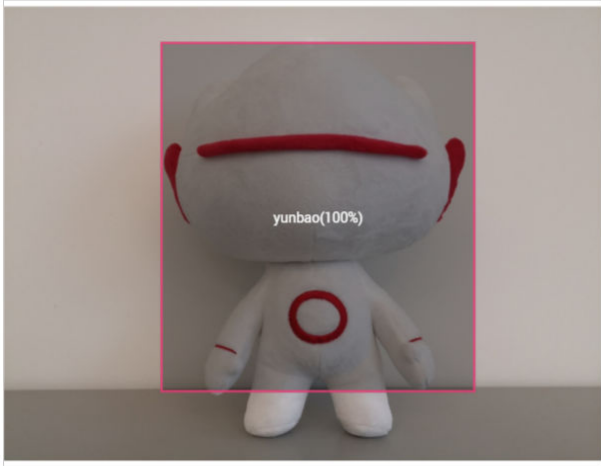
- You can also choose **Service Deployment > Real-Time Services** and click **Predict** in the **Operation** column of the target service for testing. The testing procedure is the same as that described in the following section. For details, see [Testing the Deployed Service](#).
- You can also use code to test a service. For details, see [Accessing Real-Time Services](#).
- The following describes the procedure for performing a service test after the object detection model is deployed as a service on the ExeML page.
 - a. After the model is deployed, click **Instance Details** in the service deployment phase to go to the service page. On the **Prediction** tab page, click **Upload** and select a local image for test.
 - b. Click **Predict** to perform the test. After the prediction is complete, the result is displayed in the **Test Result** pane on the right. If the model accuracy does not meet your expectation, add images on the **Label Data** tab page, label the images, and train and deploy the model again. [Table 1-8](#) describes the parameters in the prediction result. If you are satisfied with the model prediction result, call the API to access the real-time service as prompted. For details, see [Accessing Real-Time Services](#). Currently, only JPG, JPEG, BMP, and PNG images are supported.

Figure 1-31 Prediction result

Service Test

Prediction can be performed only when the service status is ● **Running**.

Select a file you want to use to test the service.



Test Result

yunbao 100%

```

1 {
2   "detection_classes": [
3     "yunbao"
4   ],
5   "detection_boxes": [
6     [
7       "60.892128",
8       "268.44058",
9       "652.6919",
10      "806.9064"
11    ]
12  ],
13  "detection_scores": [
14    "0.9998517"
15  ]
16 }
```

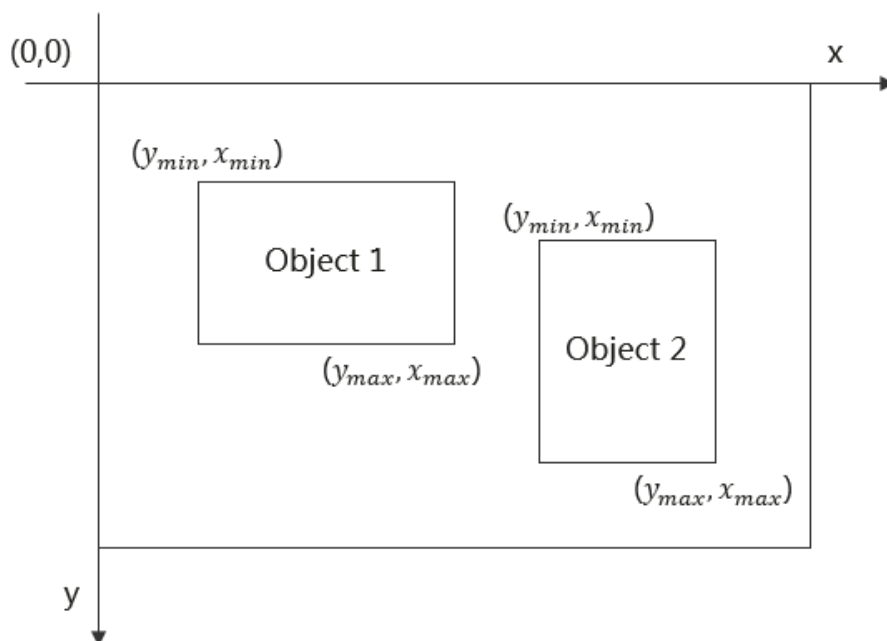
[URL](#) [API Reference](#)

<https://fa636184d4cb4042a72c0e781a3720eb.apigw.cn-north-7.mhuaweicloud.com/v1/infers/d4127df8-f44d-4263-a888-2479530b13e2>

Table 1-8 Parameters in the prediction result

Parameter	Description
detection_classes	Label of each detection box
detection_boxes	Coordinates of four points (y_min, x_min, y_max, and x_max) of each detection box, as shown in Figure 1-32
detection_scores	Confidence of each detection box

Figure 1-32 Illustration for coordinates of four points of a detection box



NOTE

- A running real-time service keeps consuming resources. If you do not need to use the real-time service, click **Stop** in the **Version Manager** pane to stop the service so that charges will no longer be incurred. If you want to use the service again, click **Start**.
- If you enable the auto stop function, the service automatically stops after the specified time and no fee is generated.

1.4 Predictive Analytics

1.4.1 Preparing Data

Before using ModelArts to build a predictive analytics model, upload data to OBS. The OBS bucket and ModelArts must be in the same region. For example, if the OBS bucket is in the CN North-Beijing4 region, ensure that the ModelArts management console is also in the CN North-Beijing4 region. Otherwise, data cannot be obtained.

Requirements on Datasets

The data set used in the predictive analytics project must be a table dataset in .csv format. For details about the table dataset, see [Creating a Dataset](#).

NOTE

To convert the data from .xlsx to .csv, perform the following operations:

Save the original table data in .xlsx. Choose **File > Save As**, select a local address, set **Save as type:** to **CSV (Comma delimited)**, and click **Save**. Then, click **OK** in the displayed dialog box.

Requirements on the training data:

- The number of columns in the training data must be the same, and there has to be at least 100 data records (a feature with different values is considered as different data records).
- The training columns cannot contain timestamp data (such as yy-mm-dd or yyyy-mm-dd).
- If a column has only one value, the column is considered invalid. Ensure that there are at least two values in the label column and no data is missing.

NOTE

The label column is the training target specified in a training task. It is the output (prediction item) for the model trained using the dataset.

- In addition to the label column, the dataset must contain at least two valid feature columns. Ensure that there are at least two values in each feature column and that the percentage of missing data must be lower than 10%.
- Due to the limitation of the feature filtering algorithm, place the predictive data column at the last. Otherwise, the training may fail.

Example of a table dataset:

The following table takes the bank deposit predictive dataset as an example. Data sources include age, occupation, marital status, cultural level, and whether there is a personal mortgage or personal loan.

Table 1-9 Fields and meanings of data sources

Field	Meaning	Type	Description
attr_1	Age	Int	Age of the customer
attr_2	Occupation	String	Occupation of the customer
attr_3	Marital status	String	Marital status of the customer
attr_4	Education status	String	Education status of the customer
attr_5	Real estate	String	Housing situation of the customer
attr_6	Loan	String	Loan of the customer
attr_7	Deposit	String	Deposit of the customer

Table 1-10 Sample data of the dataset

attr_1	attr_2	attr_3	attr_4	attr_5	attr_6	attr_7
31	blue-collar	married	secondary	yes	no	no

attr_1	attr_2	attr_3	attr_4	attr_5	attr_6	attr_7
41	management	married	tertiary	yes	yes	no
38	technician	single	secondary	yes	no	no
39	technician	single	secondary	yes	no	yes
39	blue-collar	married	secondary	yes	no	no
39	services	single	unknown	yes	no	no

Uploading Data to OBS

In this section, the OBS console is used to upload data.

Upload files to OBS according to the following specifications:

The OBS path of the predictive analytics projects must comply with the following rules:

- The OBS path of the input data must redirect to the data files. The data files must be stored in a folder in an OBS bucket rather than the root directory of the OBS bucket, for example, `/obs-xxx/data/input.csv`.
- There must be at least 100 lines of valid data in .csv. There cannot be more than 200 columns of data and the total data size must be smaller than 100 MB.

Procedure for uploading data to OBS:

Perform the following operations to import data to the dataset for model training and building.

1. Log in to the OBS console and [create a bucket](#) in the same region as ModelArts. If an available bucket exists, ensure that the OBS bucket and ModelArts are in the same region.
2. [Upload the local data](#) to the OBS bucket. If you have a large amount of data, use OBS Browser+ to upload data or folders. The uploaded data must meet the dataset requirements of the ExeML project.

NOTE

Upload data from unencrypted buckets. Otherwise, training will fail because data cannot be decrypted.

Creating a Dataset

After the data is prepared, create a proper dataset. For details, see [Creating a Dataset](#).

FAQs

How do I process Schema information when creating a table dataset using data selected from OBS?

Schema information includes the names and types of table columns, which must be the same as those of the imported data.

- If the original table contains a table header, enable **Contain Table Header**. The first row of the file will be used as column names. You do not need to modify the Schema information.
- If the original table does not contain a table header, disable **Contain Table Header**. After data is selected from OBS, the column names will be used as the first row of the table by default. Change the column names to **attr_1**, **attr_2**, ..., **attr_n**. **attr_n** is the prediction column placed at last.

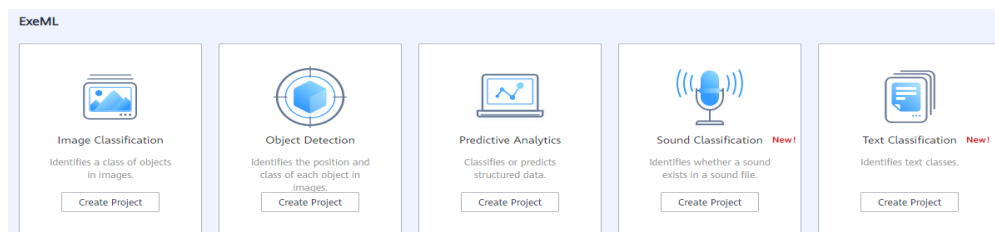
1.4.2 Creating a Project

ModelArts ExeML supports sound classification, text classification, image classification, predictive analytics, and object detection projects. You can create any of them based on your needs. Perform the following operations to create an ExeML project.

Procedure

1. Log in to the ModelArts console. In the navigation pane, choose **ExeML**.
2. Click **Create Project** in the box of your desired project.

Figure 1-33 Creating a project (1)



3. On the displayed page, set the parameters by referring to [Table 1-11](#). The default billing mode is **Pay-per-use**.

Figure 1-34 Creating a project (2)

★ Billing Mode Pay-per-use

★ Name

Description
0/500

★ Datasets Create Dataset

★ Label Column

★ Output Path Select

★ Training Flavor

Table 1-11 Parameters

Parameter	Description
Name	Name of a project <ul style="list-style-type: none"> Enter a maximum of 64 characters. Only digits, letters, underscores (_), and hyphens (-) are allowed. This parameter is mandatory. Start with a letter. The name must be unique.
Description	Brief description of a project
Datasets	You can select a dataset or click Create Dataset to create one. <ul style="list-style-type: none"> Existing dataset: Select a dataset from the drop-down list box. Only datasets of the same type are displayed. Creating a dataset: Click Create Dataset to create a dataset. For details, see Creating a Dataset.

Parameter	Description
Label Column	Select the column you want to predict. The label column is the output of an inference model. During model training, all information is used to train an inference model. The model uses the data of other columns as the input and outputs the inference result in the label column. You can publish the model as a real-time inference service.
Output Path	Select an OBS path for storing ExeML data. NOTE The output path stores all data generated in the ExeML project.
Training Flavor	Select a training flavor for this ExeML project. You will be billed based on different flavors.

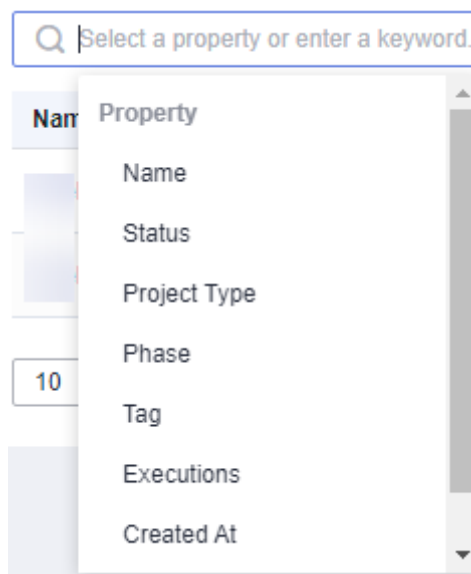
4. Click **Create Project**. Then, the ExeML workflow is displayed.
5. Wait until the workflow of the predictive analytics project executes the following phases in sequence:
 - a. **Publish Dataset Version**: Publish a version for the labeled dataset.
 - b. **Check Data**: Check whether any exception occurs in your dataset.
 - c. **Predict**: Train the dataset of the published version to generate a model.
 - d. **Register Model**: Register the trained model with model management.
 - e. **Deploy Service**: Deploy the generated model as a real-time service.

Quickly Searching for a Project

On the ExeML overview page, you can use the search box to quickly search for and filter workflows based on the ExeML type (or project name).

1. Log in to the ModelArts console. In the navigation pane, choose **ExeML**.
2. In the search box above the ExeML project list, filter the desired workflows based on the required property, such as name, status, project type, current phase, and tag.

Figure 1-35 Property




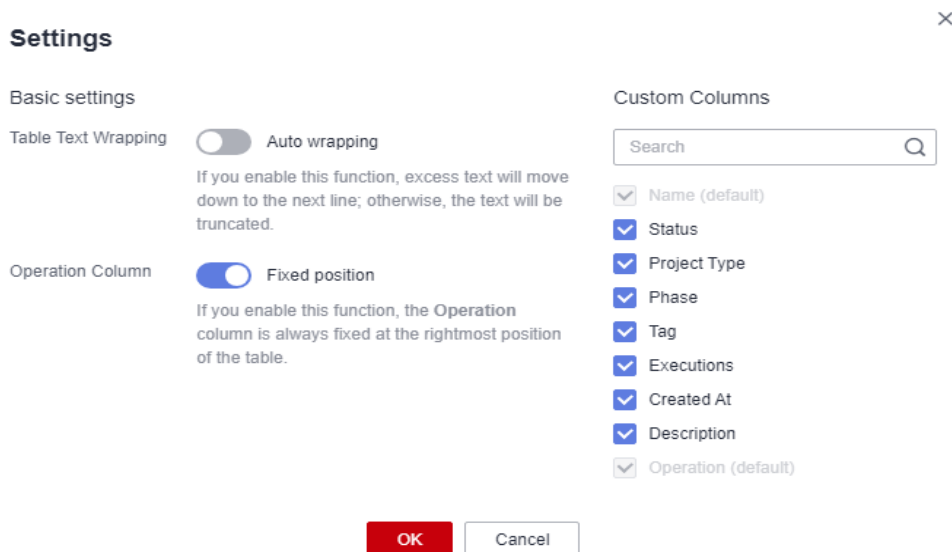
- To adjust the basic settings of ExeML and select the columns you want to see, click  on the right of the search box.

Table Text Wrapping: This function is disabled by default. If you enable this function, excess text will move down to the next line; otherwise, the text will be truncated.


Operation Column: This function is enabled by default. If you enable this function, the **Operation** column is always fixed at the rightmost position of the table.

Custom Columns: By default, all items are selected. You can select columns you want to see.

Figure 1-36 Customizing table columns



- Click **OK**. Then, the columns will be displayed based on the settings.


- To arrange ExeML projects by a specific property, click  in the table header.

1.4.3 Training a Model

After the ExeML task is created, a model is trained for predictive analytics. You can publish the model as a real-time inference service.

Procedure

- On the ExeML page of the new version, click the name of the target project to view the execution status of the current workflow.
- On the predictive analytics phase, wait until the phase status changes from **Running** to **Executed**.

- Click  to view the training details, such as the label column, data type, accuracy, and evaluation result.

The example is a discrete value of binary classification. For details about the evaluation result parameters, see [Table 1-12](#).

For details about the evaluation results generated for different data types of label columns, see [Evaluation Results](#).

NOTE

An ExeML project supports multiple rounds of training, and each round generates an AI application version. For example, the first training version is **0.0.1**, and the next version is **0.0.2**. The trained models can be managed by training version. After the trained model meets your requirements, deploy the model as a service.

Evaluation Results

The parameters in evaluation results vary depending on the training data type.

- Discrete values
The evaluation parameters include recall, precision, accuracy, and F1 score, which are described in the following table.

Table 1-12 Parameters in discrete value evaluation results

Parameter	Description
Recall	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish positive samples.
Precision	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish negative samples.
Accuracy	Fraction of correctly predicted samples over all samples. It shows the general ability of a model to recognize samples.

Parameter	Description
F1 Score	Harmonic average of the precision and recall of a model. It is used to evaluate the quality of a model. A high F1 score indicates a good model.

- Continuous values
The evaluation parameters include Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). The three error values represent a difference between a real value and a predicted value. During multiple rounds of modeling, a group of error values is generated for each round of modeling. Use these error values to determine the quality of a model. A smaller error value indicates a better model.

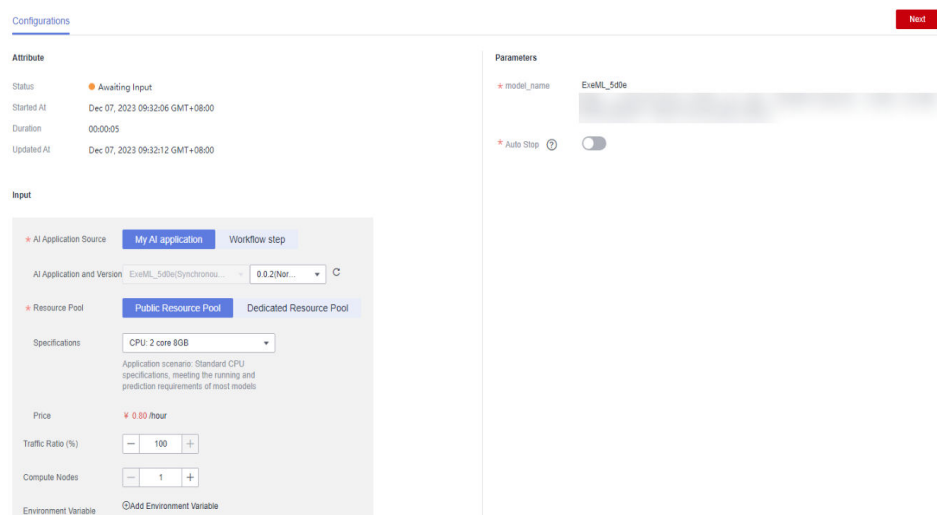
1.4.4 Deploying a Model as a Service

Deploying a Service

You can deploy a model as a real-time service that provides a real-time test UI and monitoring capabilities. After the model is trained, you can deploy a **Successful** version with ideal accuracy as a service. The procedure is as follows:

1. On the phase execution page, after the service deployment status changes to **Awaiting input**, double-click **Deploy Service**. On the configuration details page, configure resource parameters.
2. On the service deployment page, select the resource specifications used for service deployment.

Figure 1-37 Resource specifications



- **AI Application Source:** defaults to the generated AI application.
- **AI Application and Version:** The current AI application version is automatically selected, which is changeable.

- **Resource Pool:** defaults to public resource pools.
- **Traffic Ratio:** defaults to **100** and supports a value range of 0 to 100.
- **Specifications:** Select available specifications based on the list displayed on the console. The specifications in gray cannot be used in the current environment. If there are no specifications after you select a public resource pool, no public resource pool is available in the current environment. In this case, use a dedicated resource pool or contact the administrator to create a public resource pool.
- **Compute Nodes:** an integer ranging from 1 to 5. The default value is **1**.
- **Auto Stop:** enables a service to automatically stop at a specified time. If this function is disabled, a real-time service will continue to run and charges will continue to be incurred. The auto stop function is enabled by default. The default value is **1 hour later**.

The options are **1 hour later**, **2 hours later**, **4 hours later**, **6 hours later**, and **Custom**. If you select **Custom**, enter any integer from 1 to 24 in the text box on the right.

 **NOTE**

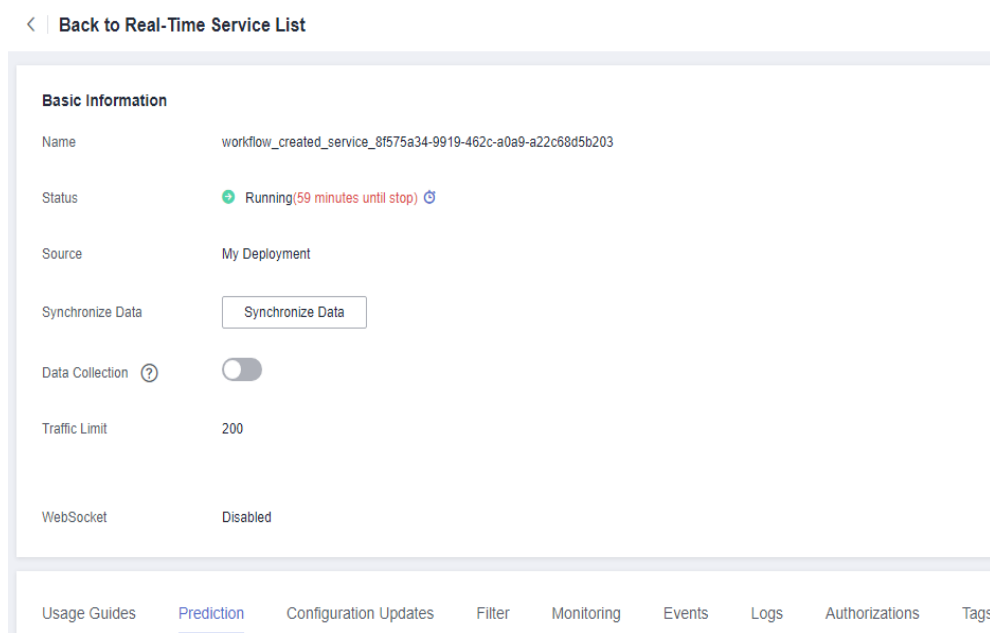
You can choose the package that you have bought when you select specifications. On the configuration fee tag, you can view your remaining package quota and how much you will pay for any extra usage.

3. After configuring resources, click **Next** and confirm the operation. Wait until the status changes to **Executed**, which means the AI application has been deployed as a real-time service.

Testing the Service

- After the service is deployed, click **Instance Details** to go to the real-time service details page. Click the **Prediction** tab to test the service.

Figure 1-38 Testing the service



- You can also choose **Service Deployment > Real-Time Services** and click **Predict** in the **Operation** column of the target service for testing. The testing procedure is the same as that described in the following section. For details, see [Testing the Deployed Service](#).
- You can also use code to test a service. For details, see [Accessing Real-Time Services](#).
- The following describes the procedure for performing a service test after the predictive analytics model is deployed as a service on the ExeML page.
 - a. After the model is deployed, you can test the model using code. In ExeML, click **Instance Details** on the **Deploy Service** page to go to the real-time service page. On the **Prediction** tab page, enter the debugging code in the **Inference Code** area.
 - b. Click **Predict** to perform the test. After the prediction is complete, the result is displayed in the **Test Result** pane on the right. If the model accuracy does not meet your expectation, train and deploy the model again on the **Label Data** tab page. If you are satisfied with the model prediction result, call the API to access the real-time service as prompted. For details, see [Accessing Real-Time Services](#).
 - In the input code, the label column of a predictive analytics database must be named **class**. Otherwise, the prediction will fail.

```
{
  "data": {
    "req_data": [{
      "attr_1": "34",
      "attr_2": "blue-collar",
      "attr_3": "single",
      "attr_4": "tertiary",
      "attr_5": "no",
      "attr_6": "tertiary"
    }]
  }
}
```

- In the preceding code snippet, **predict** is the inference result of the label column.

Figure 1-39 Prediction result

<pre>1 { 2 "data": { 3 "req_data": [{ 4 "attr_1": "34", 5 "attr_2": "blue-collar", 6 "attr_3": "single", 7 "attr_4": "tertiary", 8 "attr_5": "no", 9 "attr_6": "tertiary" 10 }] 11 } 12 }</pre>	<pre>1 { 2 "data": { 3 "resp_data": [4 { 5 "predict": "no" 6 } 7] 8 } 9 }</pre>
---	---

NOTE

- A running real-time service continuously consumes resources. If you do not need to use the real-time service, stop the service to stop billing. To do so, click **Stop** in the **More** drop-down list in the **Operation** column. If you want to use the service again, click **Start**.
- If you enable auto stop, the service automatically stops at the specified time and no fees will be generated then.

1.5 Sound Classification

1.5.1 Preparing Data

Before using ModelArts ExeML to build a model, upload data to an OBS bucket. The OBS bucket and ModelArts must be in the same region.

Requirements for Sound Classification Data

- Only 16-bit WAV files are supported. All sub-formats of WAV are supported.
- The audio must be longer than 1 second and the file must be no larger than 4 MB.
- Add more sound files to improve model precision. Prepare at least 20 sound files for each class. Ensure that the total duration of each class is no shorter than 5 minutes.
- Ensure that the sound files are authentic and cover all scenarios in real life.
- The quality of the training set has a great impact on the precision of the model. Set the sampling rate to the precision of the training set.
- The labeling quality has a great impact on the model precision. Do not mislabel objects.
- Only Chinese and English are supported for audio labeling.

Uploading Data to OBS

In this section, the OBS console is used to upload data.

Upload files to OBS according to the following specifications:

- If you do not need to upload training data in advance, create an empty folder to store files generated in the future, for example, **/bucketName/data-cat**.
- If you need to upload training data in advance, create an empty folder, and save the sound files to be labeled in the folder, for example, **/bucketName/data-cat/cat.wav**.

Procedure for uploading data to OBS:

Perform the following operations to import data to the dataset for model training and building.

1. Log in to the OBS console and **create a bucket** in the same region as ModelArts. If an available bucket exists, ensure that the OBS bucket and ModelArts are in the same region.
2. **Upload the local data** to the OBS bucket. If you have a large amount of data, use OBS Browser+ to upload data or folders. The uploaded data must meet the dataset requirements of the ExeML project.

NOTE

- Upload data from unencrypted buckets. Otherwise, training will fail because data cannot be decrypted.
- Training sound files must be classified into at least two classes, and each class must contain at least 20 sound files.

Creating a Dataset

After the data preparation is completed, create a dataset of the type supported by the project. For details, see [Creating a Dataset](#).

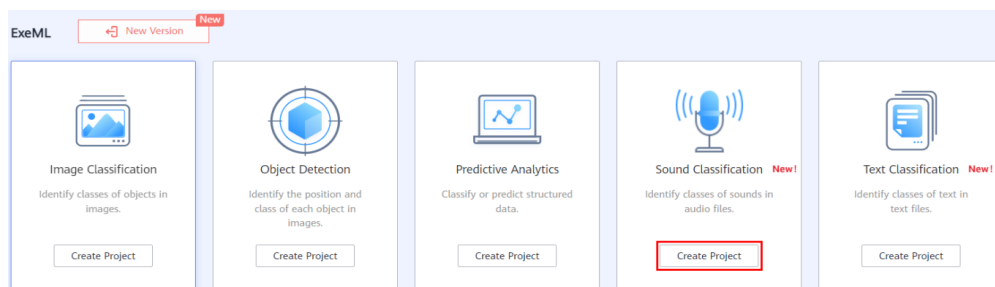
1.5.2 Creating a Project

ModelArts ExeML supports sound classification, text classification, image classification, predictive analytics, and object detection projects. You can create any of them based on your needs. Perform the following operations to create an ExeML project.

Procedure

1. Log in to the ModelArts console. In the navigation pane, choose **ExeML**.
2. Click **Create Project** in the box of your desired project. The page for creating an ExeML project is displayed.

Figure 1-40 Create Project



3. On the displayed page, configure parameters by referring to [Table 1-13](#). The default billing mode is **Pay-per-use**.

* Billing Mode Pay-per-use

* Name

Description
0/500

* Datasets Select a dataset. ↻ Create Dataset

* Output Path Select

* Training Flavor Select a flavor.

Table 1-13 Parameters

Parameter	Description
Name	Name of a project <ul style="list-style-type: none"> Enter a maximum of 64 characters. Only digits, letters, underscores (_), and hyphens (-) are allowed. This parameter is mandatory. Start with a letter. The name must be unique.
Description	Brief description of a project
Dataset Source	You can select a dataset or click Create Dataset to create one. <ul style="list-style-type: none"> Existing dataset: Select a dataset from the drop-down list box. Only datasets of the same type are displayed. Creating a dataset: Click Create Dataset to create a dataset. For details, see Creating a Dataset.
Output Path	An OBS path for storing ExeML data NOTE The output path stores all data generated in the ExeML project.
Training Flavor	Select a training flavor for this ExeML project. You will be billed based on different flavors.

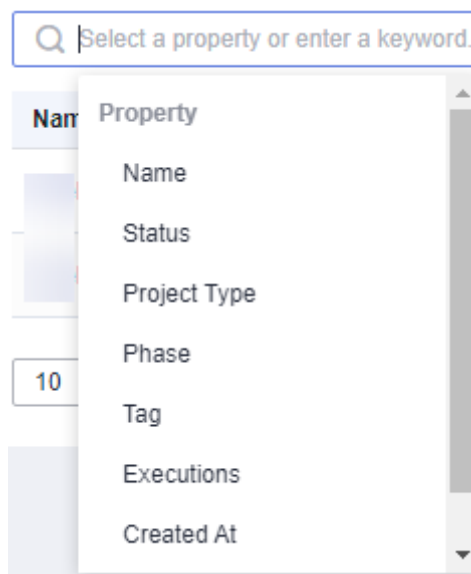
4. Click **Create Project**. Then, the ExeML workflow is displayed.
5. Wait until the workflow of the sound classification project executes the following phases in sequence:
 - a. **Label Data**: Check data labeling.
 - b. **Publish Dataset Version**: Publish a version for the labeled dataset.
 - c. **Check Data**: Check whether any exception occurs in your dataset.
 - d. **Classify Sounds**: Train the dataset of the published version to generate a model.
 - e. **Register Model**: Register the trained model with model management.
 - f. **Deploy Service**: Deploy the generated model as a real-time service.

Quickly Searching for a Project

On the ExeML overview page, you can use the search box to quickly search for and filter workflows based on the ExeML type (or project name).

1. Log in to the ModelArts console. In the navigation pane, choose **ExeML**.
2. In the search box above the ExeML project list, filter the desired workflows based on the required property, such as name, status, project type, current phase, and tag.

Figure 1-41 Property




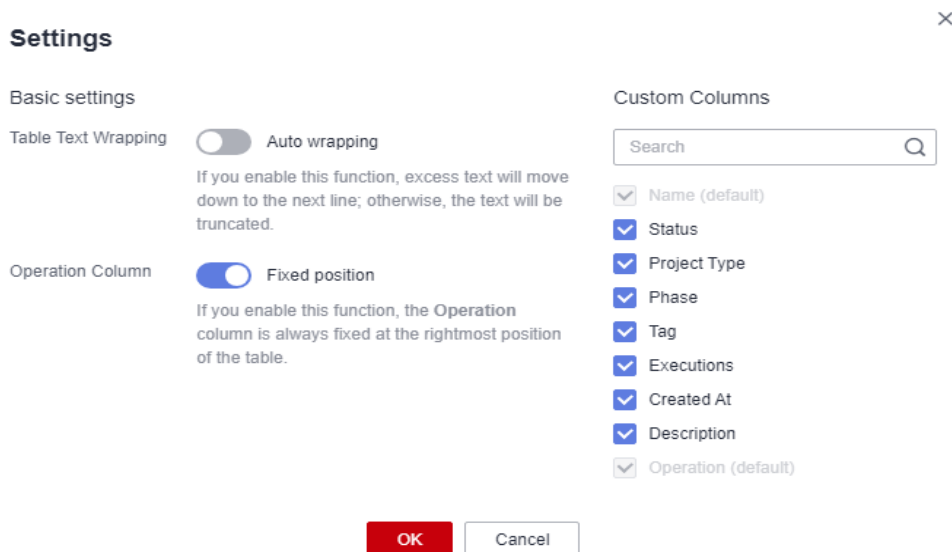
- To adjust the basic settings of ExeML and select the columns you want to see, click  on the right of the search box.

Table Text Wrapping: This function is disabled by default. If you enable this function, excess text will move down to the next line; otherwise, the text will be truncated.


Operation Column: This function is enabled by default. If you enable this function, the **Operation** column is always fixed at the rightmost position of the table.

Custom Columns: By default, all items are selected. You can select columns you want to see.

Figure 1-42 Customizing table columns



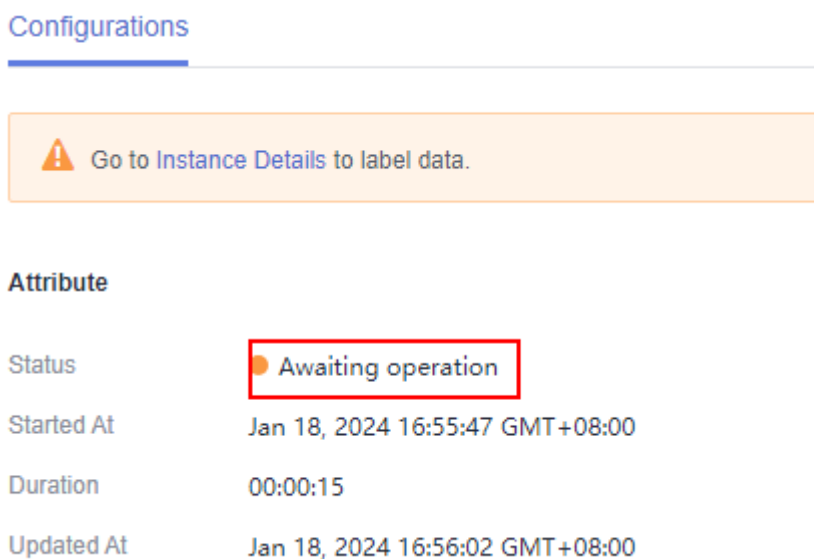
- Click **OK**. Then, the columns will be displayed based on the settings.

- To arrange ExeML projects by a specific property, click  in the table header.

1.5.3 Labeling Data

After a project is created, you will be redirected to the new-version ExeML and the project starts to run. When the data labeling phase changes to **Awaiting operation**, manually confirm data labeling in the dataset. You can also add or delete data in the dataset and modify labels.

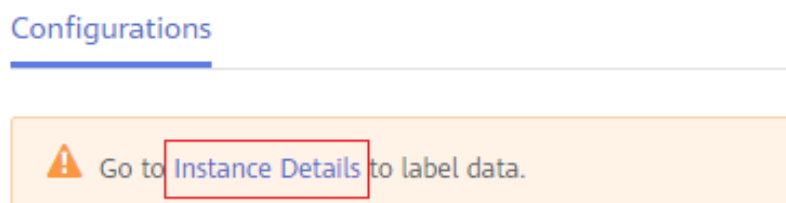
Figure 1-43 Data labeling status



Labeling Sound Files

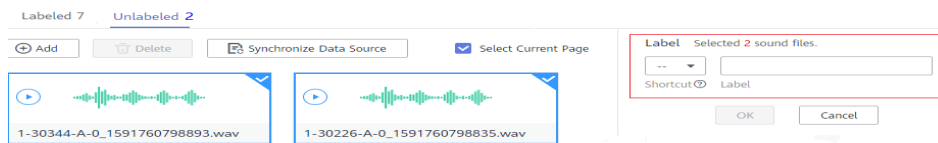
- On the labeling phase of the new-version ExeML, click **Instance Details**. The data labeling page is displayed. Click an image to go to the labeling page.

Figure 1-44 Instance Details



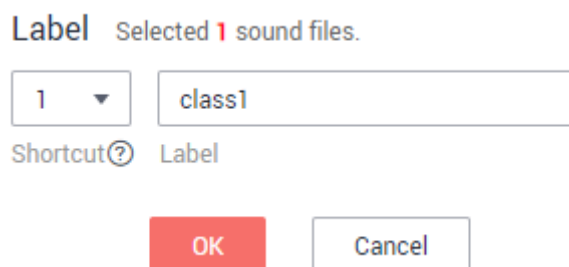
- On the labeling page, click the **Unlabeled** tab. All unlabeled sound files are displayed. Select the sound files to be labeled in sequence, or tick **Select Current Page** to select all sound files on the page, and then add labels to the sound files in the right pane.

Figure 1-45 Labeling a sound file



3. Add a label. Play a sound file, and select the sound file. In the **Label** area, enter a label name or select an existing label from the drop-down list on the right, and select a shortcut from the drop-down list on the left. Click **OK**. The selected sound file is labeled.

Figure 1-46 Adding a sound classification label



4. After all the sound files are labeled, view them on the **Labeled** tab page or view the list of **All Labels** in the right pane to learn the name and quantity of the labels.

Synchronizing or Adding Sound Files

On the data labeling phase, click **Instance Details**. The labeling page is displayed. When creating a sound classification project, you can select local data or synchronize data in OBS as the training data.

- **Add Audio:** You can quickly add sound files on a local PC to ModelArts and synchronize the files to the OBS path specified during project creation. Click **Add data** to import data.

NOTE

Only 16-bit WAV files are supported. The size of a sound file cannot exceed 4 MB. The total size of all sound files uploaded in one attempt cannot exceed 8 MB.

- **Synchronize Data Source:** To quickly obtain the latest sound files in the OBS bucket, click **Synchronize Data Source** to add sound files in OBS to ModelArts.
- **Delete Audio:** You can delete sound files one by one, or tick **Select Current Page** to delete all sound files on the page.

NOTE

The deleted sound files cannot be recovered. Exercise caution when performing this operation.

Modifying Labeled Data

After labeling data, you can modify the labeled data on the **Labeled** tab page.

- **Modifying based on audio**

On the dataset details page, click the **Labeled** tab. Select one or more audio files to be modified from the audio list. Modify the label in the label details area on the right.

- Modifying a label: In the **File Labels** area, click the editing icon in the **Operation** column, enter the correct label name in the text box, and click the check mark icon.
- Deleting a label: In the **File Labels** area, click the deletion icon in the **Operation** column. In the displayed dialog box, click **OK**.

- **Modifying based on labels**

On the data labeling page, click **Label Management** on the right. You will see information about all labels.

Figure 1-47 Information about all labels

Label Name	Attribute	Operation
bird	--	Modify Delete

- Modifying a label: Click the edit button in the **Operation** column. In the dialog box that appears, enter the new label name, select the new shortcut, and click **OK**. After the modification, the new label applies to the audio files that contain the original label.
- Deleting a label: Click the delete button in the **Operation** column. In the dialog box that appears, confirm the operation and click **OK**.

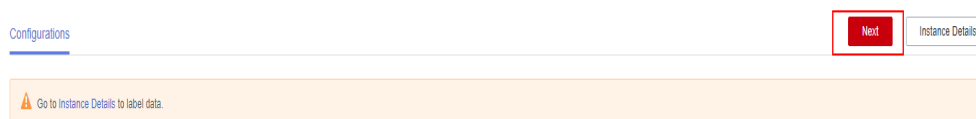
NOTE

Deleted tags cannot be restored.

Resuming Workflow Execution

After confirming data labeling, return back to the new-version ExeML. Click **Next**. Then, the workflow continues to run in sequence until all phases are executed.

Figure 1-48 Resuming the workflow execution



1.5.4 Training a Model

After labeling the sound files, train a model. You can perform model training to obtain the required sound classification model. Training sound files must be classified into at least two classes, and each class must contain at least five sound files.

Procedure

Before starting the training, label data and then perform auto training.

1. On the ExeML page of the new version, click the name of the target project. Then, click **Instance Details** on the labeling phase to label data.

Figure 1-49 Finding unlabeled files




2. Return to the labeling phase of the new-version ExeML, click **Next** and wait until the workflow enters the training phase.
3. Wait until the training is complete. No manual operation is required. If you close or exit the page, the system continues training until it is complete.
4. On the sound classification phase, wait until the training status changes from **Running** to **Executed**.
5. After the training, click  on the sound classification phase to view metric information.

Table 1-14 Evaluation result parameters

Parameter	Description
Recall	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish positive samples.
Precision	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish negative samples.
Accuracy	Fraction of correctly predicted samples over all samples. It shows the general ability of a model to recognize samples.
F1 Score	Harmonic average of the precision and recall of a model. It is used to evaluate the quality of a model. A high F1 score indicates a good model.

 **NOTE**

An ExeML project supports multiple rounds of training, and each round generates an AI application version. For example, the first training version is **0.0.1**, and the next version is **0.0.2**. The trained models can be managed by training version. After the trained model meets your requirements, deploy the model as a service.

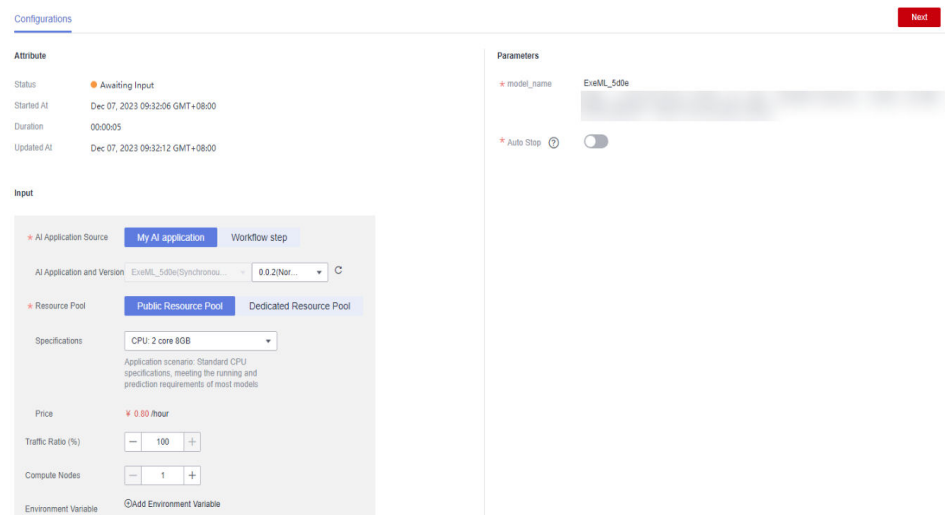
1.5.5 Deploying a Model as a Service

Deploying a Service

You can deploy a model as a real-time service that provides a real-time test UI and monitoring capabilities. After the model is trained, you can deploy a **Successful** version with ideal accuracy as a service. The procedure is as follows:

1. On the **Dashboard** page, after the service deployment status changes to **Awaiting input**, double-click **Deploy Service**. On the configuration details page, configure resource parameters.
2. On the service deployment page, select the resource specifications used for service deployment.

Figure 1-50 Resource specifications



- **AI Application Source:** defaults to the generated AI application.
- **AI Application and Version:** The current AI application version is automatically selected, which is changeable.
- **Resource Pool:** defaults to public resource pools.
- **Traffic Ratio:** defaults to **100** and supports a value range of 0 to 100.
- **Specifications:** Select available specifications based on the list displayed on the console. The specifications in gray cannot be used in the current environment. If there are no specifications after you select a public resource pool, no public resource pool is available in the current environment. In this case, use a dedicated resource pool or contact the administrator to create a public resource pool.
- **Compute Nodes:** an integer ranging from 1 to 5. The default value is **1**.
- **Auto Stop:** enables a service to automatically stop at a specified time. If this function is not enabled, the real-time service continuously runs and fees are incurred accordingly. Auto stop is enabled by default and its default value is **1 hour later**.

The auto stop options are **1 hour later**, **2 hours later**, **4 hours later**, **6 hours later**, and **Custom**. If you select **Custom**, enter any integer from 1 to 24 in the text box on the right.

 **NOTE**

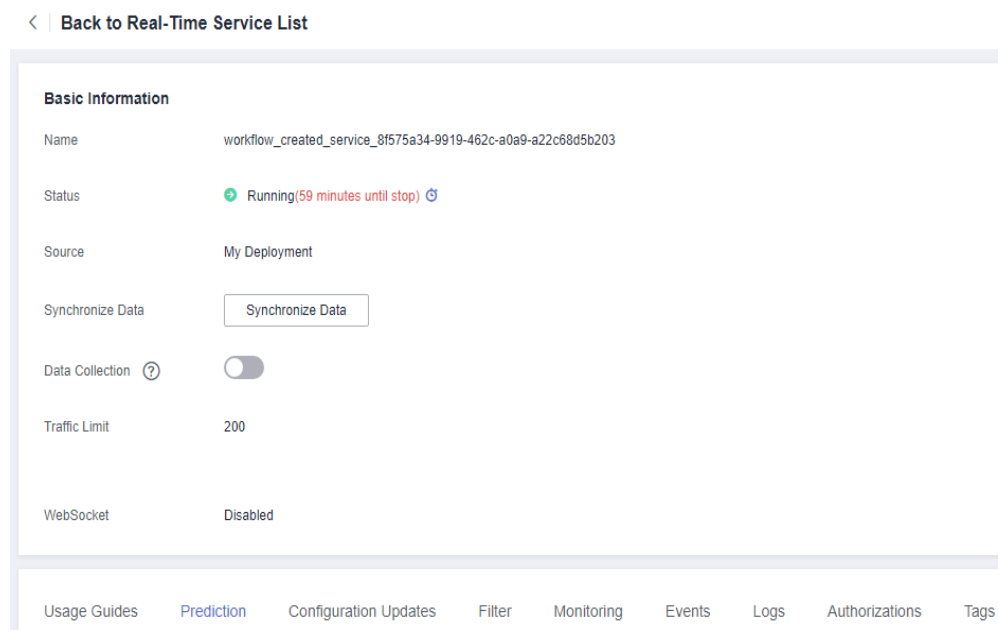
You can choose the package that you have bought when you select specifications. On the configuration fee tag, you can view your remaining package quota and how much you will pay for any extra usage.

3. After configuring resources, click **Next** and confirm the operation. Wait until the status changes to **Executed**, which means the AI application has been deployed as a real-time service.

Testing a Service

- After the service is deployed, click **Instance Details** to go to the real-time service details page. Click the **Prediction** tab to test the service.

Figure 1-51 Testing the service



- You can also choose **Service Deployment > Real-Time Services** and click **Predict** in the **Operation** column of the target service for testing. The testing procedure is the same as that described in the following section. For details, see [Testing the Deployed Service](#).
- You can also use code to test a service. For details, see [Accessing Real-Time Services](#).
- The following describes the procedure for performing a service test after the sound classification model is deployed as a service on the ExeML page.
 - a. After the model is deployed, you can add a sound file for test. On the **ExeML** page, go to the service deployment phase, click **Instance Details** to go to the **Deploy Service** tab page, select the service version in the **Running** status, click **Upload** in the service test area, and upload a local sound file to perform the test.
 - b. Click **Predict** to perform the test. After the prediction is complete, the result is displayed in the **Test Result** pane on the right. If the model accuracy does not meet your expectation, add sound files on the **Label Data** tab page, label the files, and train and deploy the model again.

Table 1-15 describes the parameters in the prediction result. If you are satisfied with the model prediction result, call the API to access the real-time service as prompted. For details, see [Accessing Real-Time Services](#).

Table 1-15 Parameters in the prediction result

Parameter	Description
predicted_label	Prediction type of the audio segment
score	Confidence score for the predicated class

 **NOTE**

A running real-time service keeps consuming resources. If you do not need to use the real-time service, click **Stop** in the **Version Manager** pane to stop the service so that charges will no longer be incurred. If you want to use the service again, click **Start**.

If you enable the auto stop function, the service automatically stops after the specified time and no fee is generated.

1.6 Text Classification

1.6.1 Preparing Data

Before using ModelArts ExeML to build a model, upload data to an OBS bucket. The OBS bucket and ModelArts must be in the same region.

Requirements on Datasets

- Files must be in TXT or CSV format, and cannot exceed 8 MB.
- Use line feed characters to separate rows in files, and each row of data represents a labeled object.
- Currently, only Chinese is supported.

Uploading Data to OBS

In this section, the OBS console is used to upload data.

Requirements for files uploaded to OBS:

- If you do not need to upload training data in advance, create an empty folder to store files generated in the future.
- If you need to upload files to be labeled in advance, create an empty folder, and save the files in the folder. An example of the file directory structure is / **bucketName/data/text.csv**.
- A label name can contain a maximum of 32 characters, including Chinese characters, letters, digits, hyphens (-), and underscores (_).
- If you want to upload labeled text files to an OBS bucket, upload them according to the following specifications:

- The objects and files to be labels must be in the same directory. The objects must be in one-to-one relationship with the files. For example, if the object file name is **COMMENTS_114745.txt**, the label file name must be **COMMENTS_114745_result.txt**.

The following shows an example of data file.

```
<dataset-import-path>
  COMMENTS_114732.txt
  COMMENTS_114732_result.txt
  COMMENTS_114745.txt
  COMMENTS_114745_result.txt
  COMMENTS_114945.txt
  COMMENTS_114945_result.txt
```

- The labeled objects and files are text files, and correspond to each other on rows. For example, the first row in the label file indicates the label of the first row in the labeled object.

Procedures for uploading data from OBS:

Perform the following operations to import data to the dataset for model training and building.

1. Log in to OBS Console and [create a bucket](#) in the same region as ModelArts. If an available bucket exists, ensure that the OBS bucket and ModelArts are in the same region.
2. [Upload the local data](#) to the OBS bucket. If you have a large amount of data, use OBS Browser+ to upload data or folders. The uploaded data must meet the dataset requirements of the ExeML project.

NOTE

- Upload data from unencrypted buckets. Otherwise, training will fail because data cannot be decrypted.
- Training text files must be classified into at least two classes, and each class must contain at least 20 rows.

Creating a Dataset

After the data preparation is completed, create a dataset of the type supported by the project. For details, see [Creating a Dataset](#).

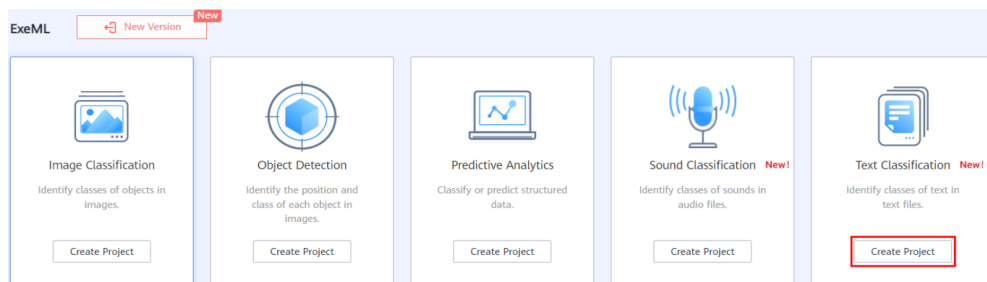
1.6.2 Creating a Project

ModelArts ExeML supports sound classification, text classification, image classification, predictive analytics, and object detection projects. You can create any of them based on your needs. Perform the following operations to create an ExeML project.

Procedure

1. Log in to the ModelArts console. In the navigation pane, choose **ExeML**.
2. Click **Create Project** in the box of your desired project. The page for creating an ExeML project is displayed.

Figure 1-52 Create Project



3. On the displayed page, configure parameters by referring to [Table 1-16](#). The default billing mode is **Pay-per-use**.

* Billing Mode Pay-per-use

* Name

Description 0/500

* Datasets ↻ Create Dataset

* Output Path Select

* Training Flavor

Table 1-16 Parameters

Parameter	Description
Name	Name of a project <ul style="list-style-type: none"> • Enter a maximum of 64 characters. Only digits, letters, underscores (_), and hyphens (-) are allowed. This parameter is mandatory. • Start with a letter. • The name must be unique.
Description	Brief description of a project

Parameter	Description
Datasets	<p>You can select a dataset or click Create Dataset to create one.</p> <ul style="list-style-type: none"> Existing dataset: Select a dataset from the drop-down list box. Only datasets of the same type are displayed. Creating a dataset: Click Create Dataset to create a dataset. For details, see "Creating a Dataset".
Output Path	<p>Select an OBS path for storing ExeML data.</p> <p>NOTE The output path stores all data generated in the ExeML project.</p>
Training Flavor	<p>Select a training flavor for this ExeML project. You will be billed based on different flavors.</p>

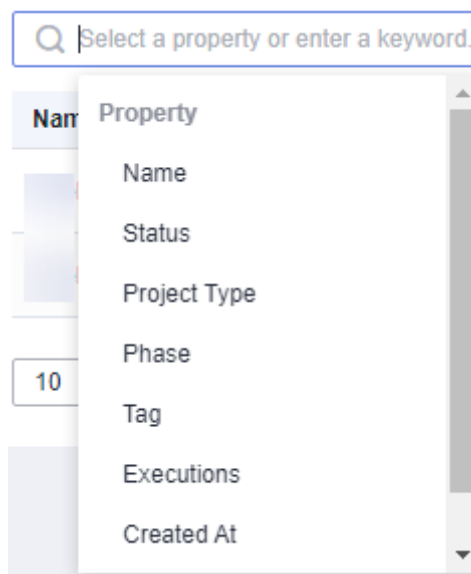
4. Click **Create Project**. Then, the ExeML workflow is displayed.
5. Wait until the workflow of the text classification project executes the following phases in sequence:
 - a. **Label Data**: Check data labeling.
 - b. **Publish Dataset Version**: Publish a version for the labeled dataset.
 - c. **Check Data**: Check whether any exception occurs in your dataset.
 - d. **Classify Text**: Train the dataset of the published version to generate a model.
 - e. **Register Model**: Register the trained model with model management.
 - f. **Deploy Service**: Deploy the generated model as a real-time service.

Quickly Searching for a Project

On the ExeML overview page, you can use the search box to quickly search for and filter workflows based on the ExeML type (or project name).

1. Log in to the ModelArts console. In the navigation pane, choose **ExeML**.
2. In the search box above the ExeML project list, filter the desired workflows based on the required property, such as name, status, project type, current phase, and tag.

Figure 1-53 Property




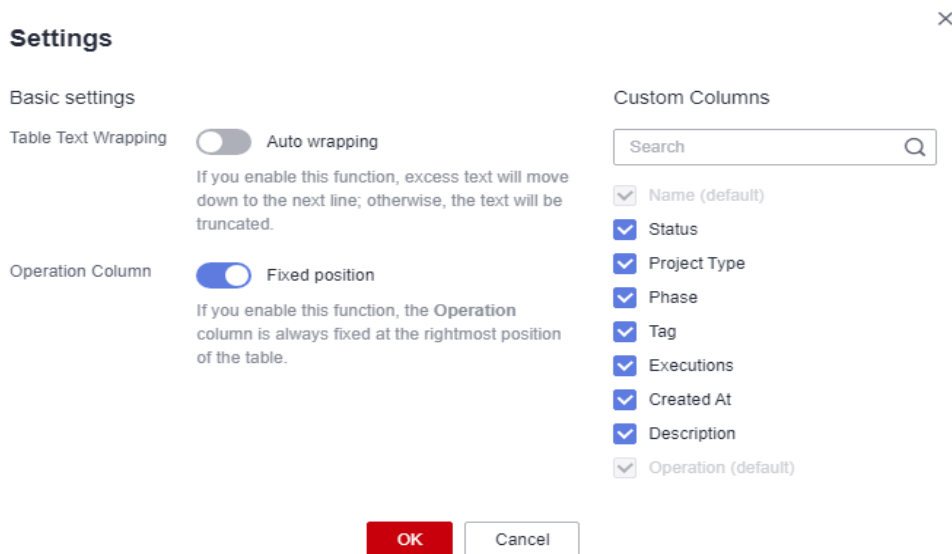
- To adjust the basic settings of ExeML and select the columns you want to see, click  on the right of the search box.

Table Text Wrapping: This function is disabled by default. If you enable this function, excess text will move down to the next line; otherwise, the text will be truncated.


Operation Column: This function is enabled by default. If you enable this function, the **Operation** column is always fixed at the rightmost position of the table.

Custom Columns: By default, all items are selected. You can select columns you want to see.

Figure 1-54 Customizing table columns



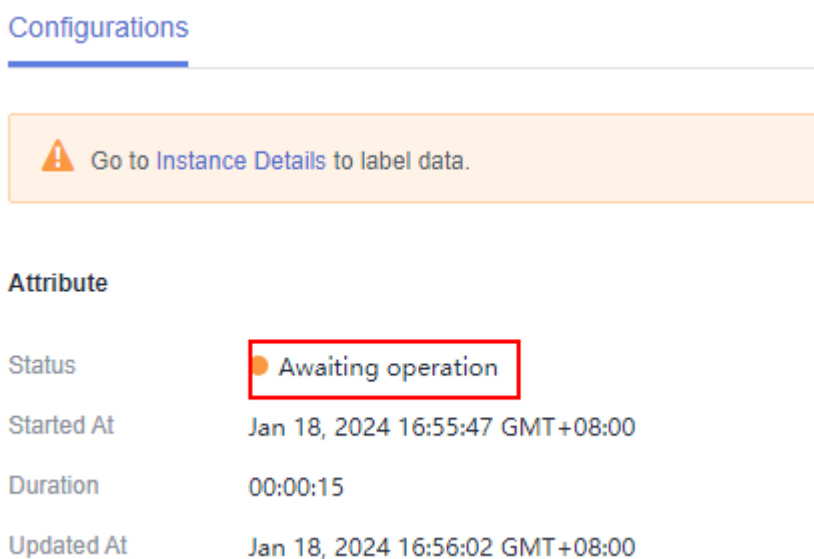
- Click **OK**. Then, the columns will be displayed based on the settings.

- To arrange ExeML projects by a specific property, click  in the table header.

1.6.3 Labeling Data

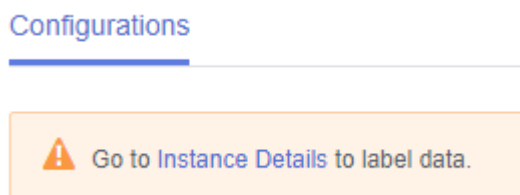
After a project is created, you will be redirected to the new-version ExeML and the project starts to run. When the data labeling phase changes to **Awaiting operation**, manually confirm data labeling in the dataset. You can also add or delete data in the dataset and modify labels.

Figure 1-55 Data labeling status



Double-click **Label Data** and click **Instance Details**. The data labeling page is displayed.

Figure 1-56 Clicking Instance Details

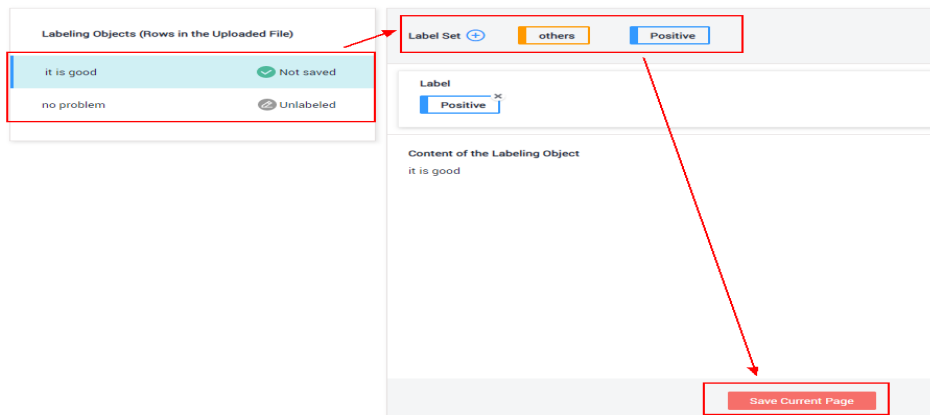


Data Labeling for Text Classification

- Select a text to be labeled in **Labeling Objects** and click different labels in the **Label Set** area to label the text.
You can add only one label for a text object.
- After confirming the file label, click **Save Current Page** in the lower right corner to save the labeling.
If a large number of objects are included in **Labeling Objects**, the page turning icon is displayed in the lower part of the area. After labeling objects

on this page, click **Save Current Page** before you turn to the next page. If you turn pages before saving the labellings, the labeling information on the previous page will be lost. You need to re-label for text data.

Figure 1-57 Data labeling - text classification



Adding or Deleting Data

In an ExeML project, the data source is the OBS directory corresponding to the input path of the dataset. If the data in the directory cannot meet your requirements, add or delete data on the ExeML page of ModelArts.

- **Adding a file**

On the **Unlabeled** tab, click **Add data** in the top left corner. In the dialog box that appears, select a local file and upload it.

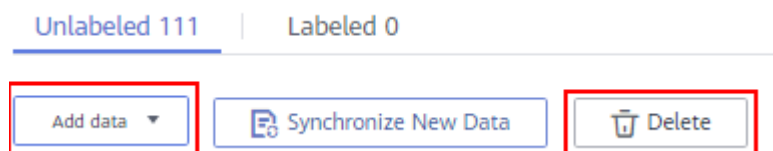
The format of the file to be uploaded must meet [requirement on datasets](#) of the text classification type.

- **Deleting a text object**

On the **Labeled** or **Unlabeled** tab page, select a text object to be deleted and click **Delete** in the upper left corner. In the dialog box that is displayed, confirm the deletion information and click **OK**.

On the **Labeled** tab page, you can tick **Select Current Page** and click **Delete** to delete all text objects and their labeling information on the current page.

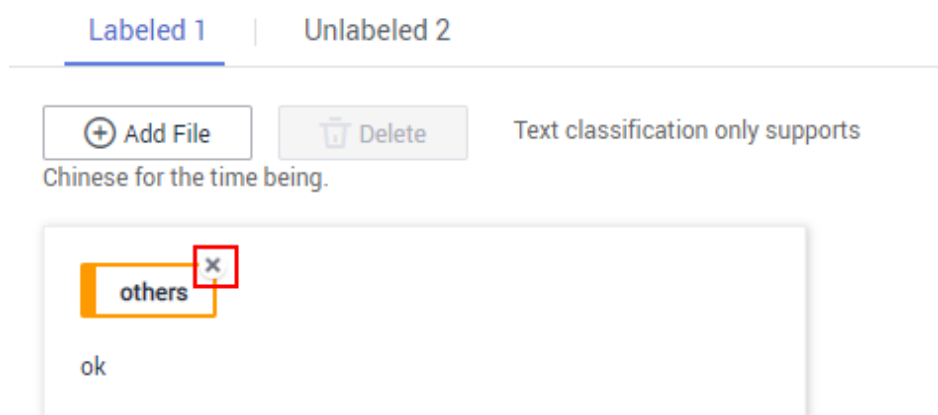
Figure 1-58 Adding a file or deleting a text object



Modifying Labeled Data

For labeled text data, only labels of the text object can be deleted. To delete a label, go to the **Labeled** tab, locate the label name area, and click the cross icon next to the label. After the label is deleted, the text object is displayed on the **Unlabeled** tab page.

Figure 1-59 Deleting a labeled text



Modifying a Label

After an ExeML project for text classification is created, you can modify labels based on service changes, including label adding, modification, and deletion.



- Adding a label
On the **Unlabeled** tab, click the plus sign (+) on the right of **Label Set**. In the **Add Label** dialog box that appears, set **Label Name** and **Label Color**, and click **OK**.
- Modifying a label
On the **Labeled** tab, locate the **All Labels** area, and click the edit button in the **Operation** column of the label you want to change. In the **Modify Label** dialog box, set **Label Name** and **Label Color** and click **OK**.
- Deleting a label
In the lower part of **All labels** on the **Labeled** tab page, select a label to be deleted and click the deletion icon in the **Operation** column. In the displayed **Delete** dialog box, select **Delete label** or **Delete the label and objects with only the label**, and click **OK**.

NOTE

The deleted labels cannot be recovered. Exercise caution when performing this operation.

Figure 1-60 Modifying a label

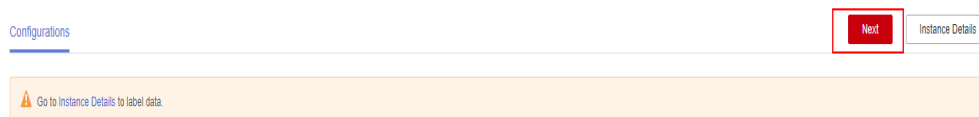
All Labels (+)

Label	Count	Operation
Positive	0	 

Resuming Workflow Execution

After confirming data labeling, return back to the new-version ExeML. Click **Next**. Then, the workflow continues to run in sequence until all phases are executed.

Figure 1-61 Resuming the workflow execution



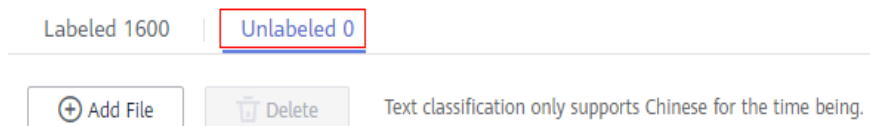
1.6.4 Training a Model

After labeling the data, train a model. You can perform model training to obtain the required text classification model. The text used for training has at least two classifications (that is, more than two labels), and the number of texts in each classification is more than 20. Before clicking **Next**, ensure that the labeled text meets the requirements.

Procedure

1. On the ExeML page of the new version, click the name of the target project. Then, click **Instance Details** on the labeling phase to label data.

Figure 1-62 Finding unlabeled files




2. Return to the labeling phase of the new-version ExeML, click **Next** and wait until the workflow enters the training phase.
3. Wait until the training is complete. No manual operation is required. If you close or exit the page, the system continues training until it is complete.
4. On the text classification phase, wait until the training status changes from **Running** to **Executed**.
5. After the training, click  on the text classification phase to view metric information. For details about the evaluation result parameters, see [Table 1-17](#).

Table 1-17 Evaluation result parameters

Parameter	Description
Recall	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish positive samples.

Parameter	Description
Precision	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish negative samples.
Accuracy	Fraction of correctly predicted samples over all samples. It shows the general ability of a model to recognize samples.
F1 Score	Harmonic average of the precision and recall of a model. It is used to evaluate the quality of a model. A high F1 score indicates a good model.

 **NOTE**

An ExeML project supports multiple rounds of training, and each round generates a version. For example, the first training version is **0.0.1**, and the next version is **0.0.2**. The trained models can be managed by training version. After the trained model meets your requirements, deploy the model as a service.

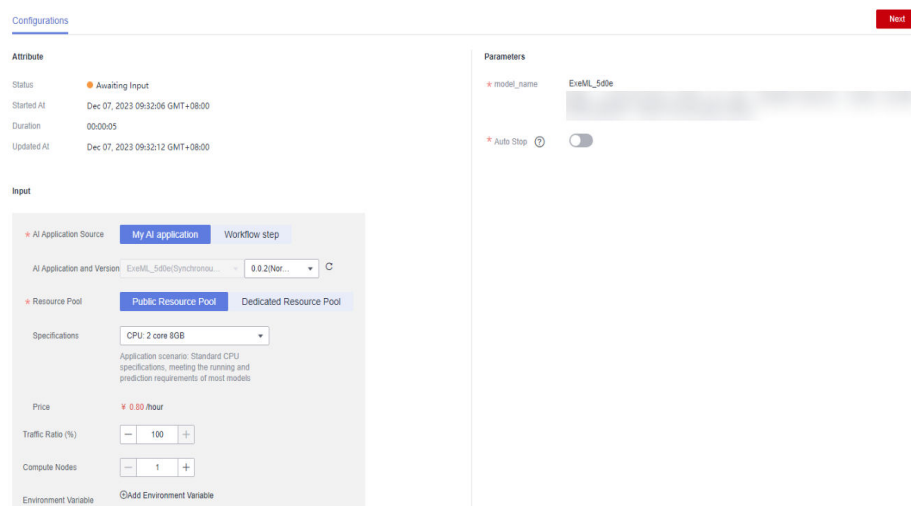
1.6.5 Deploying a Model as a Service

Deploying a Service

You can deploy a model as a real-time service that provides a real-time test UI and monitoring capabilities. After the model is trained, you can deploy a **Successful** version with ideal accuracy as a service. The procedure is as follows:

1. On the **Dashboard** page, after the service deployment status changes to **Awaiting input**, double-click **Deploy Service**. On the configuration details page, configure resource parameters.
2. On the service deployment page, select the resource specifications used for service deployment.

Figure 1-63 Resource specifications



- **AI Application Source:** defaults to the generated AI application.
- **AI Application and Version:** The current AI application version is automatically selected, which is changeable.
- **Resource Pool:** defaults to public resource pools.
- **Traffic Ratio:** defaults to **100** and supports a value range of 0 to 100.
- **Specifications:** Select available specifications based on the list displayed on the console. The specifications in gray cannot be used in the current environment. If there are no specifications after you select a public resource pool, no public resource pool is available in the current environment. In this case, use a dedicated resource pool or contact the administrator to create a public resource pool.
- **Compute Nodes:** an integer ranging from 1 to 5. The default value is **1**.
- **Auto Stop:** enables a service to automatically stop at a specified time. If this function is not enabled, the real-time service continuously runs and fees are incurred accordingly. Auto stop is enabled by default and its default value is **1 hour later**.

The auto stop options are **1 hour later**, **2 hours later**, **4 hours later**, **6 hours later**, and **Custom**. If you select **Custom**, enter any integer from 1 to 24 in the text box on the right.

 **NOTE**

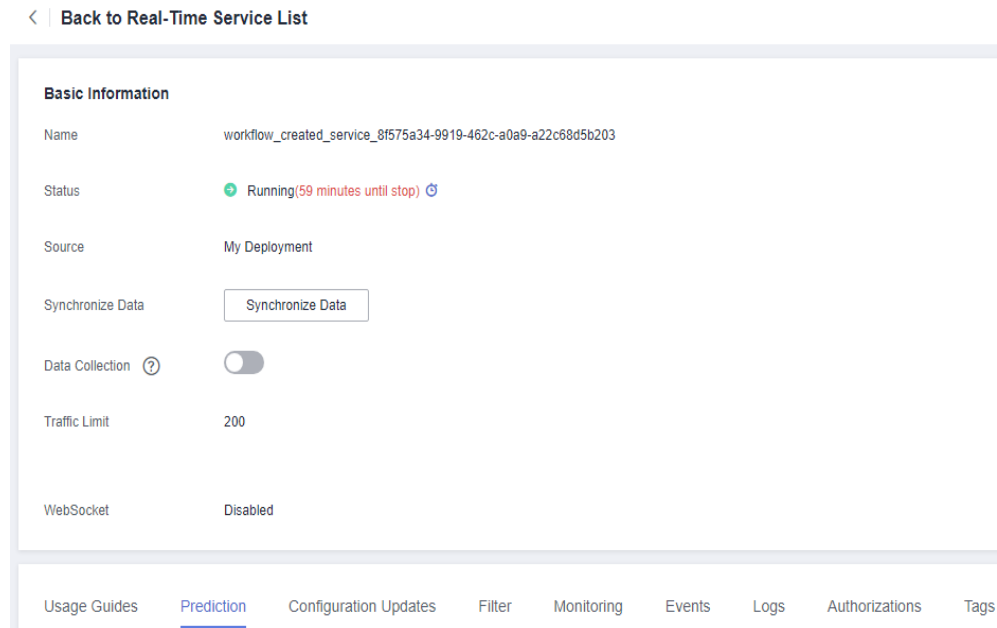
You can choose the package that you have bought when you select specifications. On the configuration fee tag, you can view your remaining package quota and how much you will pay for any extra usage.

3. After configuring resources, click **Next** and confirm the operation. Wait until the status changes to **Executed**, which means the AI application has been deployed as a real-time service.

Testing a Service

- After the service is deployed, click **Instance Details** to go to the real-time service details page. Click the **Prediction** tab to test the service.

Figure 1-64 Testing the service



- In the service deployment phase, click **Instance Details** to go to the real-time service page. Then, click **Predict** in the **Operation** column of the target service to test the service. For details, see [Testing the Deployed Service](#).
- You can also use code to test a service. For details, see [Accessing Real-Time Services](#).
- The following describes the procedure for performing a service test after the text classification model is deployed as a service on the ExeML page.
 - a. After the model is deployed, you can add a text file for test. On the **ExeML** page, click the target project, go to the **Deploy Service** tab page, select the service version in the **Running** status, and enter text to be tested in the text box in the **Service Test** area.
 - b. Click **Predict** to perform the test. After the prediction is complete, the result is displayed in the **Test Result** pane on the right. If the model accuracy does not meet your expectation, add text data files on the **Label Data** tab page, label the files, and train and deploy the model again. [Table 1-18](#) describes the parameters in the prediction result. If you are satisfied with the model prediction result, call the API to access the real-time service as prompted. For details, see [Accessing Real-Time Services](#).

Figure 1-65 Prediction

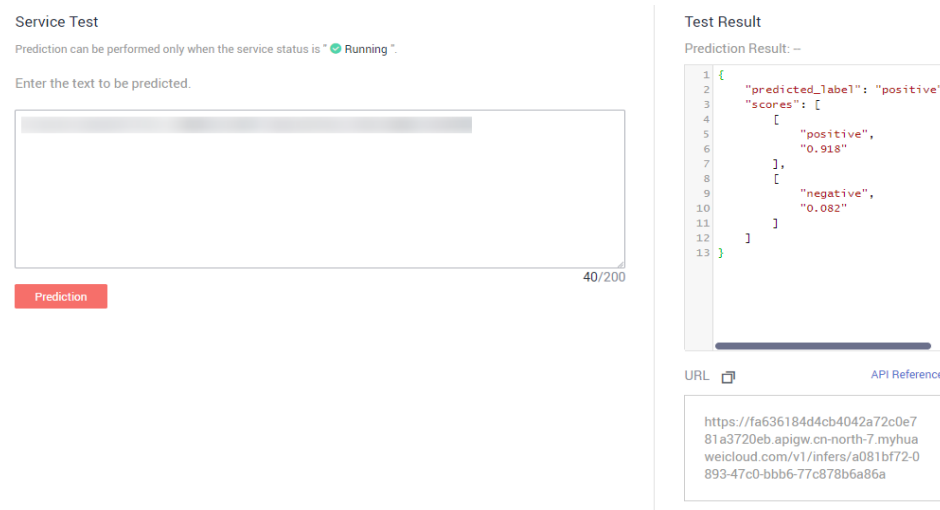


Table 1-18 Parameters in the prediction result

Parameter	Description
predicted_label	Prediction type of the text
score	Confidence score for the predicated class

NOTE

A running real-time service keeps consuming resources. If you do not need to use the real-time service, click **Stop** in the **Version Manager** pane to stop the service so that charges will no longer be incurred. If you want to use the service again, click **Start**.

If you enable the auto stop function, the service automatically stops after the specified time and no fee is generated.

1.7 Tips

1.7.1 How Do I Quickly Create an OBS Bucket and a Folder When Creating a Project?

When creating a project, select a training data path. This section describes how to quickly create an OBS bucket and folder when you select the training data path.


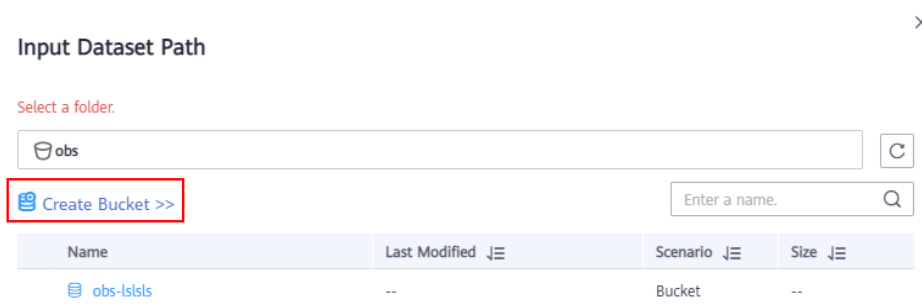
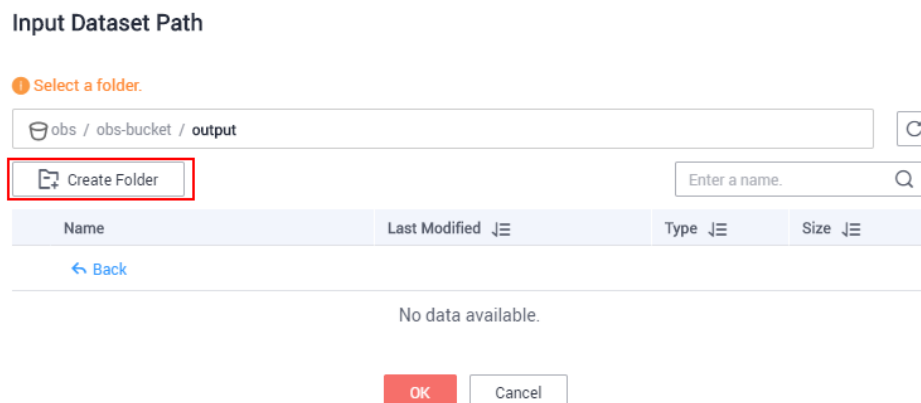
1. On the page for creating an ExeML project, click  on the right of **Input Dataset Path**. The **Input Dataset Path** dialog box is displayed.
2. Click **Create Bucket**. The **Create Bucket** page is displayed. For details, see [Creating a Bucket](#) in *Object Storage Service Console Operation Guide*.

Figure 1-66 Creating an OBS bucket



3. Select the bucket, and click **Create Folder**. In the dialog box that is displayed, enter the folder name and click **OK**.
 - The name cannot contain the following special characters: \/:*?"<>|
 - The name cannot start or end with a period (.) or slash (/).
 - The absolute path of a folder cannot exceed 1,023 characters.
 - Any single slash (/) separates and creates multiple levels of folders at once.

Figure 1-67 Creating a folder



1.7.2 Where Are Models Generated by ExeML Stored? What Other Operations Are Supported?




Unified Model Management

For an ExeML project, after the model training is complete, the generated model is automatically displayed on the **AI Application Management > AI Applications** page. See the following figure. The model name is automatically generated by the system. Its prefix is the same as the name of the ExeML project for easy identification.

CAUTION

Models generated by ExeML cannot be downloaded.

Figure 1-68 Models generated by ExeML

AI Application Name	Latest Version	Status	Deployment Type
▼ 	0.0.1	✔ Normal	Real-Time Services/Batch Ser...
▼ 	0.0.2	✔ Normal	Real-Time Services/Batch Ser...
▼ 	0.0.2	✔ Normal	Real-Time Services/Batch Ser...

What Other Operations Are Supported for Models Generated by ExeML?

- Deploying models as real-time, edge, and batch services**
 On the **ExeML** page, models can only be deployed as real-time services. You can deploy models as batch services or edge services on the **AI Application Management > AI Applications** page.
- Creating a version**
 When creating a new version, you can select a meta model only from a ModelArts training job, OBS, model template, or custom image. You cannot create a version from the original ExeML project.
- Deleting a model or its version**

2 ExeML (Old Version)

[Introduction to ExeML](#)

[Image Classification](#)

[Object Detection](#)

[Predictive Analytics](#)

[Sound Classification](#)

[Text Classification](#)

[Tips](#)

2.1 Introduction to ExeML

ExeML

ModelArts ExeML is a customized code-free model development tool that helps you start codeless AI application development with high flexibility. ExeML automates model design, parameter tuning and training, and model compression and deployment based on the labeled data. With ExeML, you only need to upload data and perform simple operations as prompted on the ExeML GUI to train and deploy models.

You can use ExeML to quickly build models for sound classification, text classification, image classification, predictive analytics, and object detection. ExeML is widely used in industrial, retail, and security sectors.

- Image classification: identifies a class of objects in images.
- Object detection: identifies the position and class of each object in an image.
- Predictive analytics: classifies or predicts structured data.
- Sound classification: classifies and identifies different sounds.
- Text classification: identifies the category of a piece of text. Currently, only Chinese is supported.

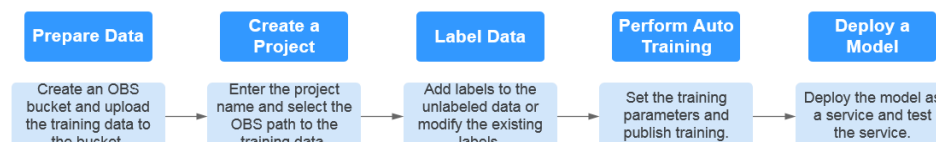
NOTE

ExeML of the old version supports only the dataset function of the old version.

ExeML Process

Figure 2-1 shows the ExeML process.

Figure 2-1 ExeML process



ExeML Projects

- **Image Classification**

An image classification project aims to classify images. You only need to add images and label them. Then, an image classification model can be quickly generated for automatically classifying offerings, vehicle types, and defective goods. For example, in the quality check scenario, you can upload a product image, label the image as qualified or unqualified, and train and deploy a model to inspect product quality.

- **Object Detection**

An object detection project aims to identify the class and location of objects in images. You only need to add images and label objects in the images with proper bounding boxes. The labeled images will be used as a training set for building a model to identify multiple objects or provide the number of objects in a single image. Object detection can also be used to inspect employees' dress code and perform unattended inspection of article placement.

- **Predictive Analytics**

A predictive analytics project is an automated model training application for structured data, which can classify or predict structured data. Predictive analytics can be used for user profile analysis and targeted marketing, as well as predictive maintenance of manufacturing equipment based on real-time data to identify equipment faults.

- **Sound Classification**

A sound classification project identifies whether a certain sound is contained in an audio file. Sound classification can be used to monitor abnormal sounds in production or security scenarios.

- **Text Classification**

A text classification project identifies the class of a piece of text.

Model Deployment Specifications

Different types of ExeML projects support different specifications for model deployment. For details, see [Table 2-1](#).

Table 2-1 Available deployment specifications for different types of projects

Project Type	Available Model Deployment Specifications
Image Classification	Free (CPU) Compute-intensive 3 instance (CPU) Compute-intensive 2 instance (GPU)
Object Detection	Free (CPU) Compute-intensive 3 instance (CPU) Compute-intensive 2 instance (GPU)
Predictive Analytics	Free (CPU) Compute-intensive 3 instance (CPU)
Sound Classification	Free (CPU) Compute-intensive 3 instance (CPU) Compute-intensive 2 instance (GPU)
Text Classification	Free (CPU) Compute-intensive 3 instance (CPU) Compute-intensive 2 instance (GPU)

2.2 Image Classification

2.2.1 Preparing Data

Before using ModelArts ExeML to build a model, upload data to an OBS bucket. The OBS bucket and ModelArts must be in the same region.

Uploading Data to OBS

This operation uses the OBS console to upload data.

Perform the following operations to import data to the dataset for model training and building.

1. Log in to OBS Console and **create a bucket** in the same region as ModelArts. If an available bucket exists, ensure that the OBS bucket and ModelArts are in the same region.
2. **Upload a file** to the OBS bucket. If you have a large amount of data, use OBS Browser+ to upload data or folders. The uploaded data must meet the dataset requirements of the ExeML project.

NOTE

Upload data from unencrypted buckets. Otherwise, training will fail because data cannot be decrypted.

Requirements on Datasets

- The name of files in a dataset cannot contain plus signs (+), spaces, or tabs.
- Ensure that no damaged image exists. The supported image formats include JPG, JPEG, BMP, and PNG.
- Do not store data of different projects in the same dataset.
- Prepare sufficient data and balance each class of data. To achieve better results, prepare at least 100 images of each class in a training set for image classification.
- To ensure the prediction accuracy of models, the training samples must be similar to the actual application scenarios.
- To ensure the generalization capability of models, datasets should cover all possible scenarios.

Requirements for Files Uploaded to OBS

- If you do not need to upload training data in advance, create an empty folder to store files generated in the future, for example, **/bucketName/data-cat**.
- If you need to upload images to be labeled in advance, create an empty folder and save the images in the folder. An example of the image directory structure is **/bucketName/data-cat/cat.jpg**.
- If you want to upload labeled images to the OBS bucket, upload them according to the following specifications:
 - The dataset for image classification requires storing labeled objects and their label files (in one-to-one relationship with the labeled objects) in the same directory. For example, if the name of the labeled object is **10.jpg**, the name of the label file must be **10.txt**.

Example of data files:

```
└─<dataset-import-path>
  10.jpg
  10.txt
  11.jpg
  11.txt
  12.jpg
  12.txt
```

- Images in JPG, JPEG, PNG, and BMP formats are supported. When uploading images on the OBS console, ensure that the size of an image does not exceed 5 MB and the total size of images to be uploaded in one attempt does not exceed 8 MB. If the data volume is large, use OBS Browser+ to upload images.
- A label name can contain a maximum of 32 characters, including letters, digits, hyphens (-), and underscores (_).
- Image classification label file (.txt) rule:
Each row contains only one label.

```
cat
dog
...
```

2.2.2 Creating a Project

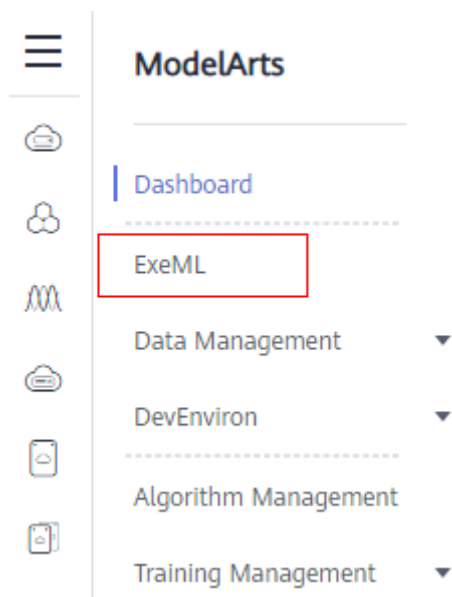
ModelArts ExeML supports sound classification, text classification, image classification, predictive analytics, and object detection projects. You can create

any of them based on your needs. Perform the following operations to create an ExeML project.

Procedure

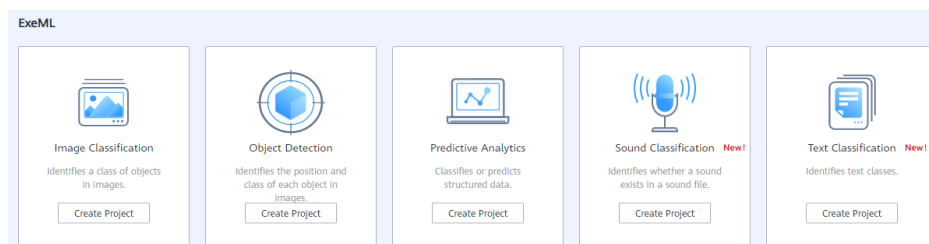
1. Log in to the ModelArts management console. In the left navigation pane, choose **ExeML**.

Figure 2-2 ExeML



2. Click **Create Project** in the box of your desired project. The page for creating an ExeML project is displayed.

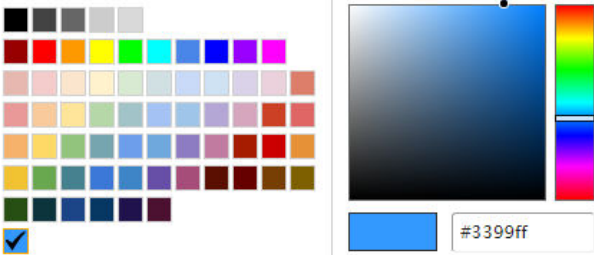
Figure 2-3 ExeML project list



3. On the displayed page, set the parameters by referring to **Table 2-2**. The default billing mode is **Pay-per-use**.

Table 2-2 Parameters

Parameter	Description
Name	<p>Name of an ExeML project</p> <ul style="list-style-type: none"> • Enter a maximum of 32 characters. Only digits, letters, underscores (_), and hyphens (-) are allowed. This parameter is mandatory. • The name must start with a letter.

Parameter	Description
Description	Brief description of a project
Dataset Source	<p>You can create a dataset or specify an existing dataset.</p> <ul style="list-style-type: none"> • Create: Configure parameters such as Dataset Name, Input Dataset Path, Output Dataset Path, and Label Set. • Specify: Select a dataset of the same type from ModelArts Data Management to create an ExeML project. Only datasets of the same type are displayed in the Dataset Name drop-down list.
Dataset Name	<p>If you select Create for Dataset Source, enter a dataset name based on required rules in the text box on the right. If you select Specify for Dataset Source, select one from available datasets of the same type under the current account displayed in the drop-down list.</p>
Input Dataset Path	<p>Select the OBS path to the input dataset. For details about dataset input specifications, see Preparing Data.</p> <ul style="list-style-type: none"> • Only the files and folders described in Preparing Data > Requirements for Files Uploaded to OBS can be saved in the training data path. Otherwise, an error will be reported. • Do not modify the files in the training data path.
Output Dataset Path	<p>Select the OBS path for storing the output dataset.</p> <p>NOTE The output dataset path cannot be the same as the input dataset path or cannot be the subdirectory of the input dataset path. It is a good practice to select an empty directory in Output Dataset Path.</p>
Label Set	<ul style="list-style-type: none"> • Label Name: Enter a label name. The label name can contain only Chinese characters, letters, digits, underscores (_), and hyphens (-), which contains 1 to 32 characters. • Add Label: Click Add Label to add one or more labels. • Set the label color: You need to set label colors for object detection and text classification datasets, but you do not need to set label colors for image and sound classification datasets. Select a color from the color palette on the right of a label, or enter the hexadecimal color code to set the color. 

4. Click **Create Project**. The system displays a message indicating that the project has been created. Then, the **Label Data** tab page is displayed. Alternatively, view the created project on the **ExeML** page and click the project name to go to the **Label Data** page.

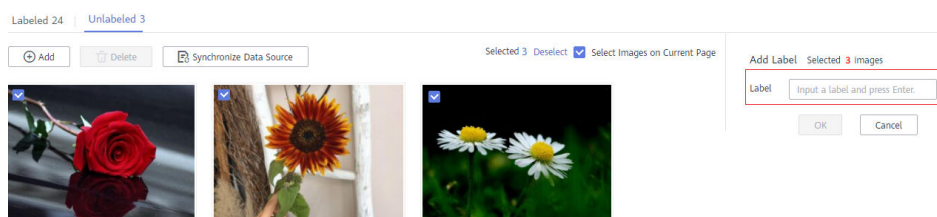
2.2.3 Labeling Data

Model training requires a large number of labeled images. Therefore, before model training, add labels to the images that are not labeled. ModelArts allows you to add labels in batches by one click. You can also modify or delete labels that have been added to images. Prepare at least two classes of images for training. Each class contains at least five images. To achieve better effect, prepare at least 50 images for each class. If the image classes are similar, more images are required.

Labeling Images

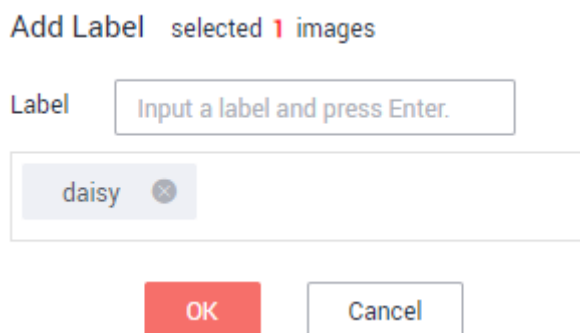
1. On the **Label Data** tab page, click the **Unlabeled** tab. All unlabeled images are displayed. Select the images to be labeled in sequence, or tick **Select Current Page** to select all images on the page, and then add labels to the images in the right pane.

Figure 2-4 Image label



2. After selecting an image, input a label in the **Label** text box, or select an existing label from the drop-down list. Click **OK**. The selected image is labeled. For example, you can select multiple images containing tulips and add label **tulips** to them. Then select other unlabeled images and label them as **sunflowers** and **roses**. After the labeling is complete, the images are saved on the **Labeled** tab page.
 - a. You can add multiple labels to an image.
 - b. A label name can contain a maximum of 32 characters, including letters, digits, hyphens (-), and underscores (_).

Figure 2-5 Image labeling



3. After all the images are labeled, view them on the **Labeled** tab page or view **All Labels** in the right pane to check the name and quantity of the labels.

Synchronizing or Adding Images

On the **ExeML** page, click the project name. The **Label Data** tab page is displayed. When creating a project, you can add images from a local PC or synchronize image data from OBS.

Figure 2-6 Adding local images

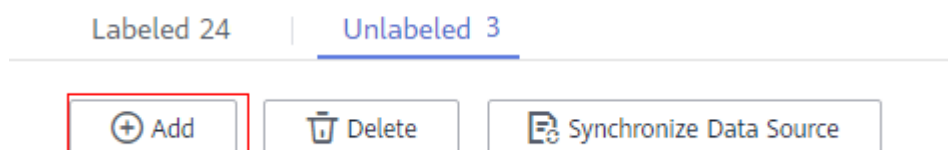
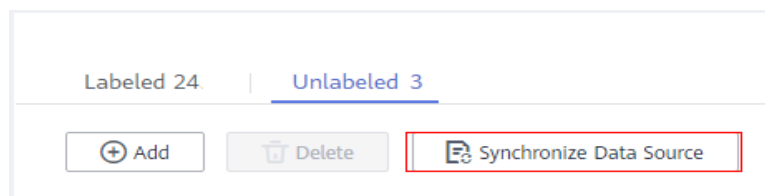


Figure 2-7 Synchronizing OBS images



- **Add:** You can quickly add images on a local PC to ModelArts. These images will be automatically synchronized to the OBS path specified during project creation. Click **Add Image**. In the dialog box that is displayed, click **Add Image** and add images. The total size of all images uploaded in one attempt cannot exceed 8 MB. The size of a single image cannot exceed 5 MB.
- **Synchronize Data Source:** You can upload images to the OBS directory specified during project creation and click **Synchronize Data Source** to quickly add the images in the OBS directory to ModelArts.
- **Delete Image:** You can delete images one by one, or tick **Select Current Page** to delete all images on the page.

NOTE

The deleted images cannot be recovered. Exercise caution when performing this operation.

Modifying Labeled Data

After labeling data, you can modify the labeled data on the **Labeled** tab page.

- **Modifying based on images**
On the data labeling page, click the **Labeled** tab, and select one or more images to be modified from the image list. Modify the image information in the label information area on the right.
 - Adding a label: In the **Label** text box, select an existing label, or enter a new label name and click **OK** to add the label to the selected image.




- Modifying a label: In the **Labels of Selected Images** area, click the editing icon in the **Operation** column, enter the correct label name in the text box, and click  to complete the modification.

Figure 2-8 Modifying a label

Labels of Selected Images		
Name	Labels	Operation
Cigarettebutts	1	 









- Deleting a label: In the **Labels of Selected Image** area, click  in the **Operation** column to delete the label.
- **Modifying based on labels**
On the dataset labeling page, click the **Labeled** tab. The information about all labels is displayed on the right.

Figure 2-9 Information about all labels

All Labels 8		
Name	Labels 	Operation
Cigarettebutts	1	 
Eggshells	100	 
fruitpeel	100	 

- Modifying a label: Click the editing icon in the **Operation** column. In the dialog box that is displayed, enter the new label name and click **OK**. After the modification, the images that have been added with the label use the new label name.
- Deleting a label: Click the delete icon in the **Operation** column. In the displayed dialog box, select **Delete the label, Delete the label and the images that only have this label, but do not delete source files, or Delete the label and the images that only have this label and also delete source files**, and click **Yes**.

Figure 2-10 Deleting a label

Delete



Are you sure you want to delete the [daisy] label?

This label contains 100 images. A deleted label cannot be recovered. Exercise caution when performing this operation.

- Delete the label
- Delete the label and the images that only have this label, but do not delete source files
- Delete the label and the images that only have this label and also delete source files ?

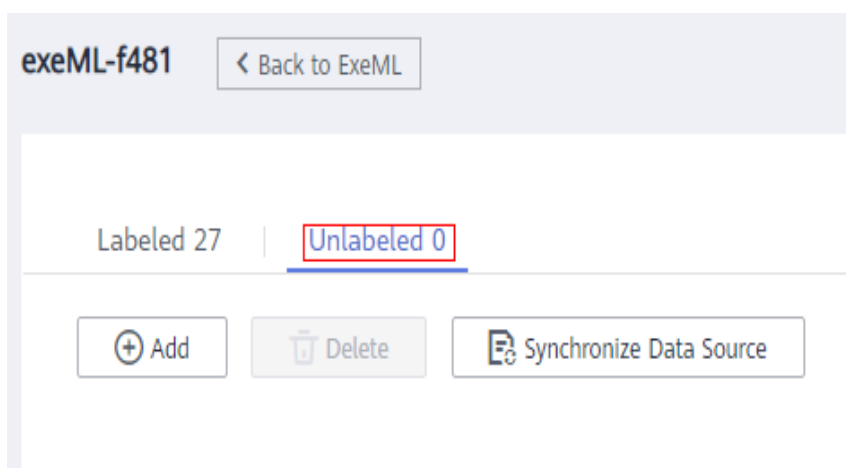
2.2.4 Training a Model

After labeling the images, you can train a model. You can perform model training to obtain the required image classification model. Training images must be classified into at least two classes, and each class must contain at least five images. Before training, ensure that the labeled images meet the requirements. Otherwise, the **Train** button is unavailable.

Procedure

1. On the **ExeML** page, click the name of the project that is successfully created. The **Label Data** tab page is displayed.

Figure 2-11 Finding unlabeled images



2. On the **Label Data** tab page, click **Train** in the upper right corner. In the displayed **Training Configuration** dialog box, set related parameters. [Table 2-3](#) describes the parameters. Ensure that the number of decimal places of training and validation ratios ranges from 1 to 5.

Figure 2-12 Setting training parameters

Training Configuration

* Dataset Version

Training and Validation Ratios ? Training Set Ratio: ?
 Validation Set Ratio: 0.2

Max Training Time (Minute)

Training Preference ? ▼

Instance Flavor ▼

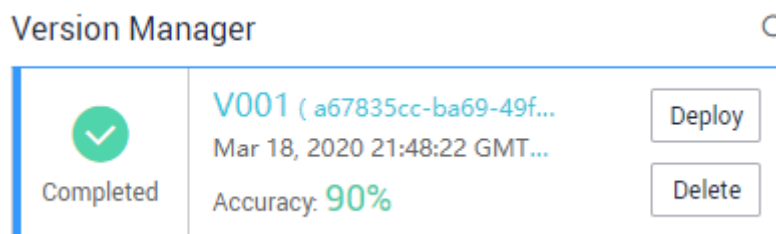
Table 2-3 Parameter description

Parameter	Description	Default Value
Dataset Version	This version is the one when the dataset is published in Data Management . In an ExeML project, when a training job is started, the dataset is published as a version based on the previous data labeling. The system automatically provides a version number. You can change it to the version number that you want.	Randomly provided by the system
Training and Validation Ratios	The labeled sample is randomly divided into a training set and a validation set. By default, the ratio for the training set is 0.8, and that for the validation set is 0.2. The usage field in the manifest file records the set type. The value ranges from 0 to 1.	0.8
Incremental Training Version	Select the version with the highest precision to perform training again. This accelerates model convergence and improves training precision.	None

Parameter	Description	Default Value
Max. Training Duration (Minute)	If training is not completed within the maximum training duration, the model is saved and training stops. To prevent the model from exiting before convergence, set this parameter to a large value. The value ranges from 6 to 6000. It is a good practice to properly extend the training duration.	60
Training Preference	<ul style="list-style-type: none"> ● performance_first: performance first. The training duration is short and the generated model is small. ● balance: balanced performance and precision ● accuracy_first: precision first. The training duration is long and the generated model is large. 	balance
Instance Flavor	<p>Select the resource specifications used for training. By default, the following specifications are supported:</p> <ul style="list-style-type: none"> ● Compute-intensive 1 instance (GPU): This flavor is billed on a pay-per-use basis. ● Free (GPU): This flavor is free. However, if the flavor is used, the training job automatically stops after one hour. That is, the training job lasts for only one hour at a time. You are advised to evaluate the data size and ensure that the time of a training job does not exceed 1 hour. When there are a large number of users, they need to wait in a queue for this flavor. <p>The compute flavors are for reference only. Obtain the flavors on the management console.</p>	Free (GPU)

3. After configuring training parameters, click **Next** to go to the configuration page, confirm the specifications, and click **Submit** to start auto model training. The training takes a certain period of time. Wait until the training is complete. If you close or exit this page, the system still performs the training operation.
To use the free flavor, read the message carefully and select **I have read and agree to the above**.
4. On the **Train Model** tab page, wait until the training status changes from **Running** to **Completed**.

Figure 2-13 Running successful



5. View the training details, such as **Accuracy**, **Evaluation Result**, **Training Parameters**, and **Classification Statistics**. For details about the evaluation result parameters, see [Table 2-4](#).

Figure 2-14 Model training result

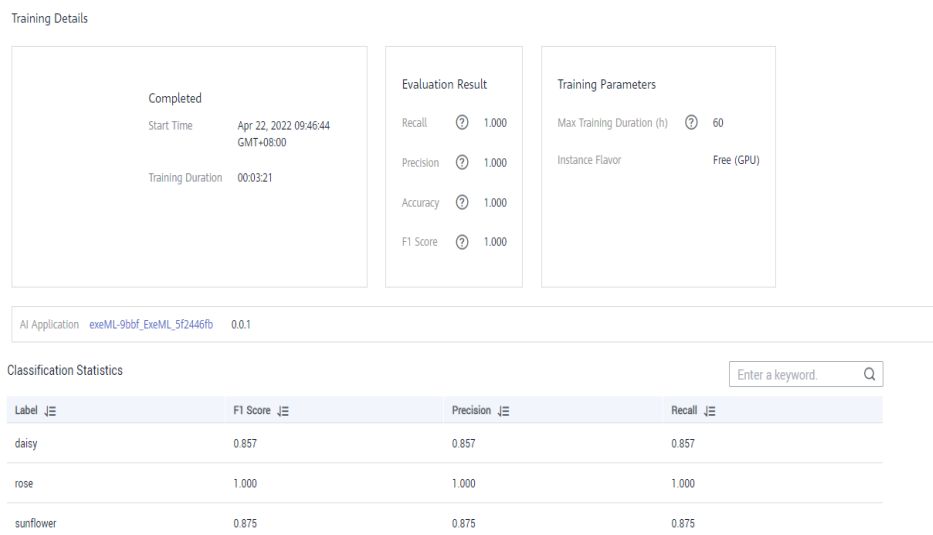


Table 2-4 Evaluation result parameters

Parameter	Description
Recall	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish positive samples.
Precision	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish negative samples.
Accuracy	Fraction of correctly predicted samples over all samples. It shows the general ability of a model to recognize samples.
F1 Score	Harmonic average of the precision and recall of a model. It is used to evaluate the quality of a model. A high F1 score indicates a good model.

NOTE

An ExeML project supports multiple rounds of training, and each round generates a version. For example, the first training version is **V001 (xxx)**, and the next version is **V002 (xxx)**. The trained models can be managed by training version. After the trained model meets your requirements, deploy the model as a service.

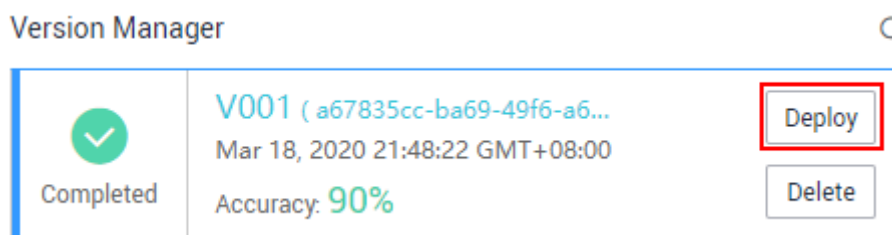
2.2.5 Deploying a Model as a Service

Deploying a Model

You can deploy a model as a real-time service that provides a real-time test UI and monitoring capabilities. After model training is complete, you can deploy a version with the ideal accuracy and in the **Successful** status as a service. The procedure is as follows:

1. On the **Train Model** tab page, wait until the training status changes to **Successful**. Click **Deploy** in the **Version Manager** pane to deploy the model as a real-time service.

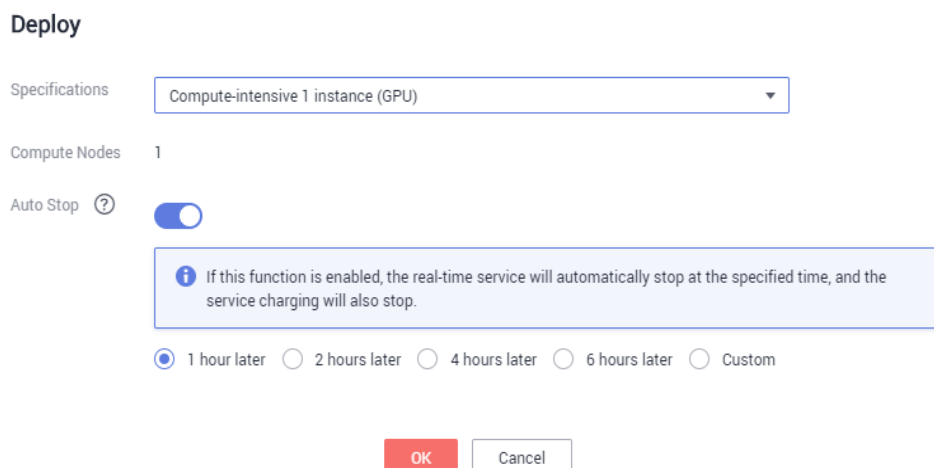
Figure 2-15 Deploy



2. In the **Deploy** dialog box, select resource flavor, set the **Auto Stop** function, and click **OK** to start the deployment.
 - **Specifications:** The GPU specifications are better, and the CPU specifications are more cost-effective.
 - **Compute Nodes:** The default value is **1** and cannot be changed.
 - **Auto Stop:** After this function is enabled and the auto stop time is set, a service automatically stops at the specified time. If this parameter is disabled, a real-time service keeps running and billing. The function can help you avoid unnecessary billing. The auto stop function is enabled by default, and the default value is **1 hour later**.

The options are **1 hour later**, **2 hours later**, **4 hours later**, **6 hours later**, and **Custom**. If you select **Custom**, you can enter any integer from 1 to 24 hours in the text box on the right.

Figure 2-16 Deploying a model



3. After the model deployment is started, view the deployment status on the **Service Deployment** page.

It takes a certain period of time to deploy a model. When the status in the **Version Manager** pane changes from **Deploying** to **Running**, the deployment is complete.

NOTE

On the **ExeML** page, trained models can only be deployed as real-time services. For details about how to deploy them as batch services or edge services, see [Where Are Models Generated by ExeML Stored? What Other Operations Are Supported?](#)

Testing a Service

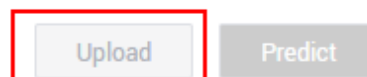
- On the **Service Deployment** page, select a service type. For example, on the ExeML page, the image classification model is deployed as a real-time service by default. On the **Real-Time Services** page, click **Prediction** in the **Operation** column of the target service to perform a service test. For details, see [Testing the Deployed Service](#).
- You can also use code to test a service. For details, see [Accessing Real-Time Services](#).
- The following describes the procedure for performing a service test after the image classification model is deployed as a service on the ExeML page.
 - a. After the model is deployed, test the service using an image. On the **ExeML** page, click the target project, go to the **Deploy Service** tab page, select the service version in the **Running** status, click **Upload** in the service test area, and upload a local image to perform the test.

Figure 2-17 Uploading an image

Service Test

Prediction can be performed only when the service status is " Running ".

Select a file you want to use to test the service.



- b. Click **Prediction** to conduct the test. After the prediction is complete, label **sunflowers** and its detection score are displayed in the prediction result area on the right. If the model accuracy does not meet your expectation, add images on the **Label Data** tab page, label the images, and train and deploy the model again. **Table 2-5** describes the parameters in the prediction result. If you are satisfied with the model prediction result, call the API to access the real-time service as prompted. For details, see **Accessing Real-Time Services**.

Currently, only JPG, JPEG, BMP, and PNG images are supported.

Figure 2-18 Prediction result

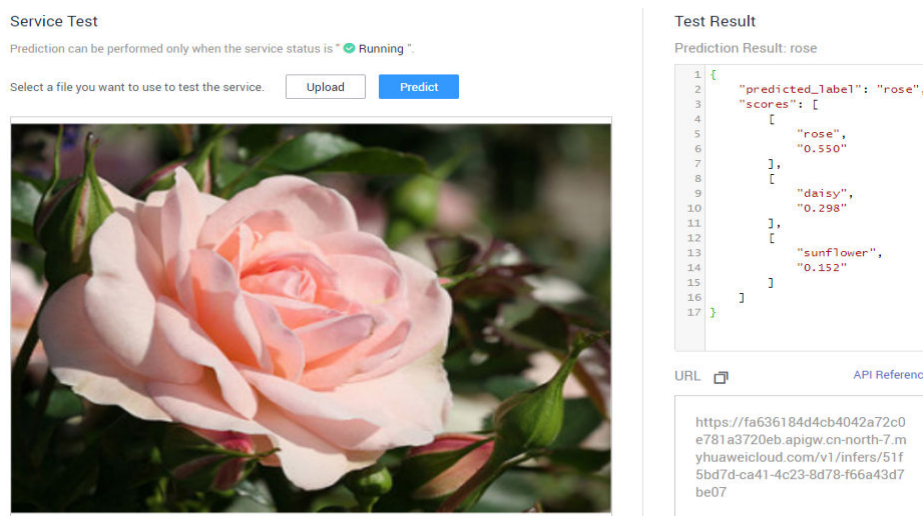


Table 2-5 Parameters in the prediction result

Parameter	Description
predict_label	Image prediction label
scores	Prediction confidence of top 5 labels

NOTE

A running real-time service keeps consuming resources. If you do not need to use the real-time service, click **Stop** in the **Version Manager** pane to stop the service so that charges will no longer be incurred. If you want to use the service again, click **Start**.

If you enable the auto stop function, the service automatically stops after the specified time and no fee is generated.

2.3 Object Detection

2.3.1 Preparing Data

Before using ModelArts ExeML to build a model, upload data to an OBS bucket. The OBS bucket and ModelArts must be in the same region.

Uploading Data to OBS

This operation uses the OBS console to upload data.

Perform the following operations to import data to the dataset for model training and building.

1. Log in to OBS Console and **create a bucket** in the same region as ModelArts. If an available bucket exists, ensure that the OBS bucket and ModelArts are in the same region.
2. **Upload a file** to the OBS bucket. If you have a large amount of data, use OBS Browser+ to upload data or folders. The uploaded data must meet the dataset requirements of the ExeML project.

NOTE

Upload data from unencrypted buckets. Otherwise, training will fail because data cannot be decrypted.

Requirements on Datasets

- The name of files in a dataset cannot contain Chinese characters, plus signs (+), spaces, or tabs.
- Ensure that no damaged image exists. The supported image formats include JPG, JPEG, BMP, and PNG.
- Do not store data of different projects in the same dataset.
- To ensure the prediction accuracy of models, the training samples must be similar to the actual application scenarios.
- To ensure the generalization capability of models, datasets should cover all possible scenarios.
- In an object detection dataset, if the coordinates of the bounding box exceed the boundaries of an image, the image cannot be identified as a labeled image.

Requirements for Files Uploaded to OBS

- If you do not need to upload training data in advance, create an empty folder to store files generated in the future, for example, **/bucketName/data-cat**.
- If you need to upload images to be labeled in advance, create an empty folder and save the images in the folder. An example of the image directory structure is **/bucketName/data-cat/cat.jpg**.
- If you want to upload labeled images to the OBS bucket, upload them according to the following specifications:
 - The dataset for object detection requires storing labeled objects and their label files (in one-to-one relationship with the labeled objects) in the same directory. For example, if the name of the labeled object is **IMG_20180919_114745.jpg**, the name of the label file must be **IMG_20180919_114745.xml**.

The label files for object detection must be in PASCAL VOC format. For details about the format, see [Table 2-6](#).

Example of data files:

```

|<dataset-import-path>
|   IMG_20180919_114732.jpg
|   IMG_20180919_114732.xml
|   IMG_20180919_114745.jpg
|   IMG_20180919_114745.xml
|   IMG_20180919_114945.jpg
|   IMG_20180919_114945.xml

```

- Images in JPG, JPEG, PNG, and BMP formats are supported. When uploading images on the OBS console, ensure that the size of an image does not exceed 5 MB and the total size of images to be uploaded in one attempt does not exceed 8 MB. If the data volume is large, use OBS Browser+ to upload images.
- A label name can contain a maximum of 32 characters, including letters, digits, hyphens (-), and underscores (_).

Table 2-6 PASCAL VOC format description

Field	Mandatory	Description
folder	Yes	Directory where the data source is located
filename	Yes	Name of the file to be labeled
size	Yes	Image pixel <ul style="list-style-type: none"> • width: image width. This parameter is mandatory. • height: image height. This parameter is mandatory. • depth: number of image channels. This parameter is mandatory.
segmented	Yes	Segmented or not

Field	Mandatory	Description
object	Yes	<p>Object detection information. Multiple object{} functions are generated for multiple objects.</p> <ul style="list-style-type: none"> • name: class of the labeled object. This parameter is mandatory. • pose: shooting angle of the labeled object. This parameter is mandatory. • truncated: whether the labeled object is truncated (0 indicates that the object is not truncated). This parameter is mandatory. • occluded: whether the labeled object is occluded (0 indicates that the object is not occluded). This parameter is mandatory. • difficult: whether the labeled object is difficult to identify (0 indicates that the object is easy to identify). This parameter is mandatory. • confidence: confidence score of the labeled object. The value range is 0 to 1. This parameter is optional. • bndbox: bounding box type. This parameter is mandatory. For details about the possible values, see Table 2-7.

Table 2-7 Description of bounding box types

type	Shape	Labeling Information
bndbox	Rectangle	<p>Coordinates of the upper left and lower right points</p> <pre><xmin>100<xmin> <ymin>100<ymin> <xmax>200<xmax> <ymin>200<ymin></pre>

Example of the label file in KITTI format:

```
<annotation>
  <folder>test_data</folder>
  <filename>260730932.jpg</filename>
  <size>
    <width>767</width>
    <height>959</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>bag</name>
```

```
<pose>Unspecified</pose>  
<truncated>0</truncated>  
<occluded>0</occluded>  
<difficult>0</difficult>  
<bndbox>  
  <xmin>108</xmin>  
  <ymin>101</ymin>  
  <xmax>251</xmax>  
  <ymax>238</ymax>  
</bndbox>  
</object>  
</annotation>
```

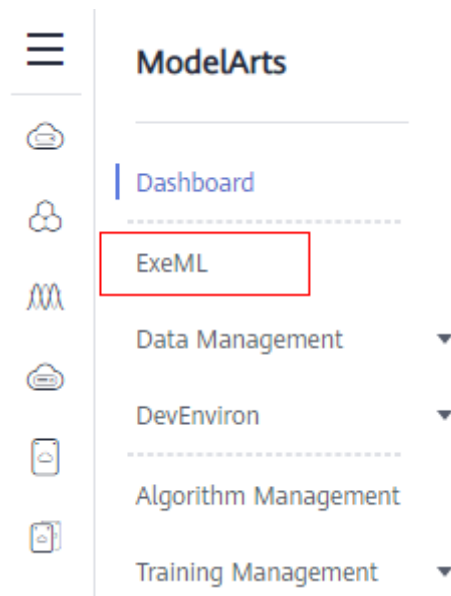
2.3.2 Creating a Project

ModelArts ExeML supports sound classification, text classification, image classification, predictive analytics, and object detection projects. You can create any of them based on your needs. Perform the following operations to create an ExeML project.

Procedure

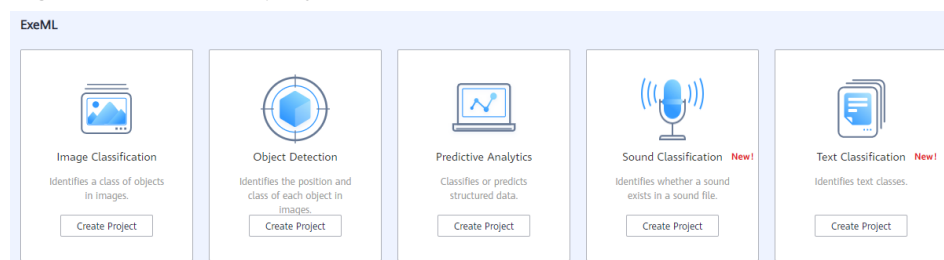
1. Log in to the ModelArts management console. In the left navigation pane, choose **ExeML**.

Figure 2-19 ExeML



2. Click **Create Project** in the box of your desired project. The page for creating an ExeML project is displayed.

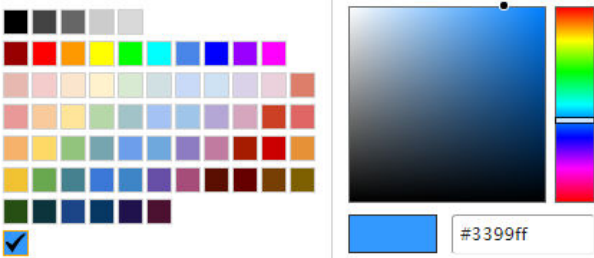
Figure 2-20 ExeML project list



- On the displayed page, set the parameters by referring to [Table 2-8](#). The default billing mode is **Pay-per-use**.

Table 2-8 Parameters

Parameter	Description
Name	<p>Name of an ExeML project</p> <ul style="list-style-type: none"> Enter a maximum of 32 characters. Only digits, letters, underscores (_), and hyphens (-) are allowed. This parameter is mandatory. The name must start with a letter.
Description	Brief description of a project
Dataset Source	<p>You can create a dataset or specify an existing dataset.</p> <ul style="list-style-type: none"> Create: Configure parameters such as Dataset Name, Input Dataset Path, Output Dataset Path, and Label Set. Specify: Select a dataset of the same type from ModelArts Data Management to create an ExeML project. Only datasets of the same type are displayed in the Dataset Name drop-down list.
Dataset Name	<p>If you select Create for Dataset Source, enter a dataset name based on required rules in the text box on the right. If you select Specify for Dataset Source, select one from available datasets of the same type under the current account displayed in the drop-down list.</p>
Input Dataset Path	<p>Select the OBS path to the input dataset. For details about dataset input specifications, see Preparing Data.</p> <ul style="list-style-type: none"> Only the files and folders described in Preparing Data > Requirements for Files Uploaded to OBS can be saved in the training data path. Otherwise, an error will be reported. Do not modify the files in the training data path.
Output Dataset Path	<p>Select the OBS path for storing the output dataset.</p> <p>NOTE The output dataset path cannot be the same as the input dataset path or cannot be the subdirectory of the input dataset path. It is a good practice to select an empty directory in Output Dataset Path.</p>

Parameter	Description
Label Set	<ul style="list-style-type: none"> ● Label Name: Enter a label name. The label name can contain only Chinese characters, letters, digits, underscores (_), and hyphens (-), which contains 1 to 32 characters. ● Add Label: Click Add Label to add one or more labels. ● Set the label color: You need to set label colors for object detection and text classification datasets, but you do not need to set label colors for image and sound classification datasets. Select a color from the color palette on the right of a label, or enter the hexadecimal color code to set the color. 

4. Click **Create Project**. The system displays a message indicating that the project has been created. Then, the **Label Data** tab page is displayed. Alternatively, view the created project on the **ExeML** page and click the project name to go to the **Label Data** page.

2.3.3 Labeling Data

Before data labeling, consider how to design labels. The labels must correspond to the distinct characteristics of the detected images and are easy to identify (the detected object in an image is highly distinguished from the background). Each label specifies the expected recognition result of the detected images. After the label design is complete, prepare images based on the designed labels. It is recommended that the number of all images to be detected be greater than 100. If the labels of some images are similar, prepare more images.

- During labeling, the variance of a class should be as small as possible. That is, the labeled objects of the same class should be as similar as possible. The labeled objects of different classes should be as different as possible.
- The contrast between the labeled objects and the image background should be as stark as possible.
- In object detection labeling, a target object must be entirely contained within a labeling box. If there are multiple objects in an image, do not relabel or miss any objects.

Labeling Images

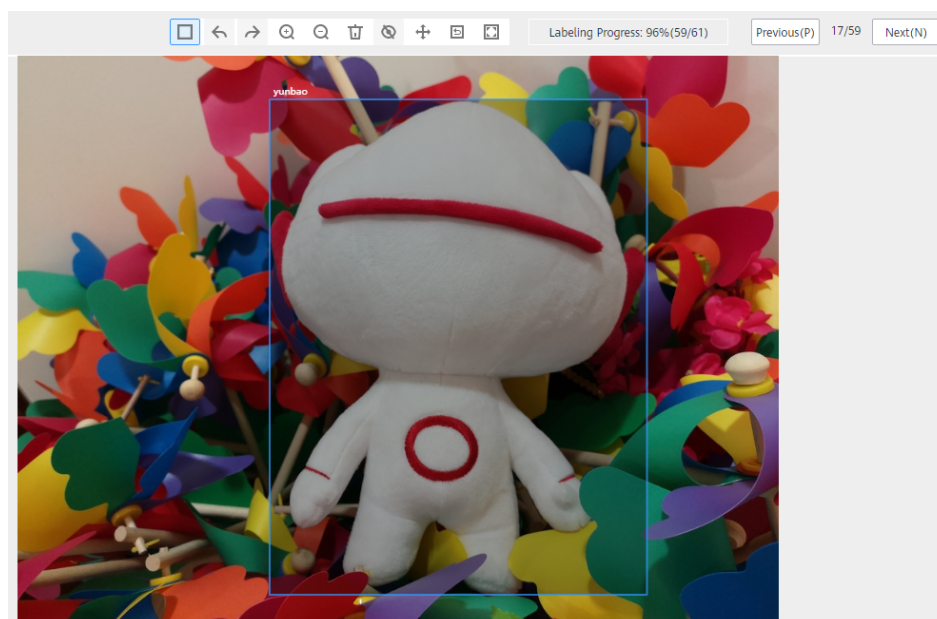
1. On the **Label Data** tab page, click the **Unlabeled** tab. All unlabeled images are displayed. Click an image to go to the labeling page.

2. Left-click and drag the mouse to select the area where the target object is located. In the dialog box that is displayed, select the label color, enter the label name, for example, **yunbao**, and press **Enter**. After the labeling is complete, the status of the images changes to **Labeled**.

More descriptions of data labeling are as follows:

- You can click the arrow keys in the upper and lower parts of the image, or press the left and right arrow keys on the keyboard to select another image. Then, repeat the preceding operations to label the image. If an image contains more than one object, you can label all the objects.
- You can add multiple labels with different colors for an object detection ExeML project for easy identification. After selecting an object, select a new color and enter a new label name in the dialog box that is displayed to add a new label.
- In an ExeML project, object detection supports only rectangular labeling boxes. In the **Data Management** function, more types of labeling boxes are supported for object detection datasets.
- In the **Label Data** window, you can scroll the mouse to zoom in or zoom out on the image to quickly locate the object.

Figure 2-21 Image labeling for object detection



3. After all images in the image directory are labeled, click **ExeML** in the upper left corner. In the dialog box that is displayed, click **OK** to save the labeling information. The **Label Data** page is displayed. On the **Labeled** tab page, you can view the labeled images or view the label names and quantity in the right pane.

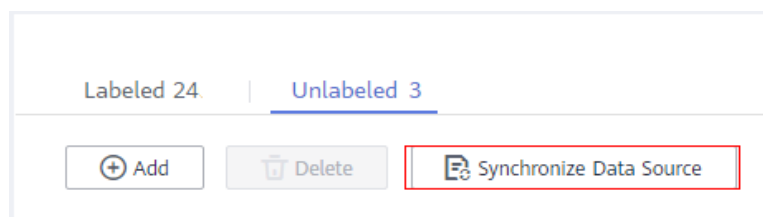
Synchronizing or Adding Images

On the **ExeML** page, click the project name. The **Label Data** tab page is displayed. When creating a project, you can add images from a local PC or synchronize image data from OBS.

Figure 2-22 Adding local images



Figure 2-23 Synchronizing OBS images



- **Add:** You can quickly add images on a local PC to ModelArts. These images will be automatically synchronized to the OBS path specified during project creation. Click **Add**. In the dialog box that is displayed, click **Add Image** and add images. The total size of all images uploaded at one time cannot exceed 8 MB. The size of a single image cannot exceed 5 MB.
- **Synchronize Data Source:** You can upload images to the OBS directory specified during project creation and click **Synchronize Data Source** to quickly add the images in the OBS directory to ModelArts.
- **Delete:** You can delete images one by one, or select **Select Images on Current Page** to delete all images on the page.

NOTE


Deleted images cannot be recovered.

Modifying Labeled Data

After labeling data, you can modify the labeled data on the **Labeled** tab page.

- **Modifying based on images**

On the dataset details page, click the **Labeled** tab, and then select the image to be modified. Modify the image information in the label information area on the right.

 - **Modifying a label:** In the **Labeling** area, click the editing icon, enter the correct label name in the text box, and click  to complete the modification. The label color cannot be modified.
 - **Deleting a label:** In the **Labeling** area, click the deletion button to delete a label for the image.

After the label is deleted, click the project name in the upper left corner of the page to exit the labeling page. The image will be returned to the **Unlabeled** tab page.

Figure 2-24 Editing an object detection label



- Modifying based on labels**
 On the dataset details page, click the **Labeled** tab. The information about all labels is displayed on the right. Click the editing icon in the **Operation** column. In the dialog box that is displayed, enter the new label name and click **OK**. After the modification, the images that have been added with the label use the new label name.

Figure 2-25 All labels for object detection

All Labels 2

Name	Labels	Samples	Operation
kitch...	0	0	
recycl...	1	1	

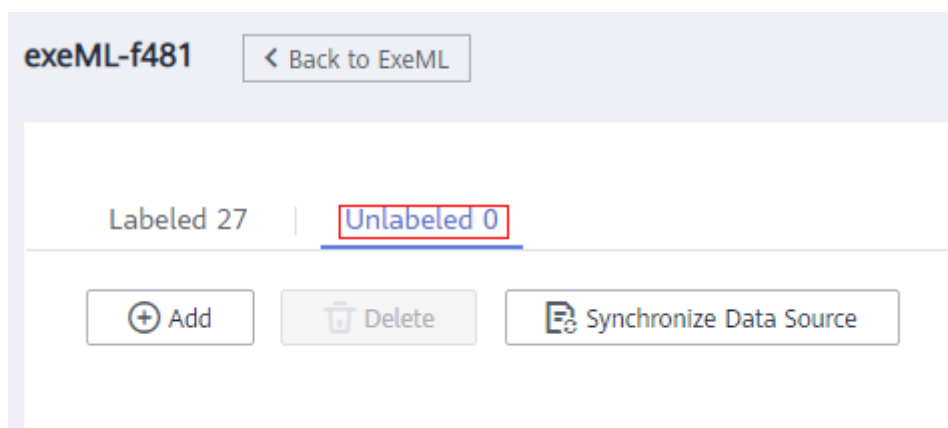
2.3.4 Training a Model

After labeling the images, perform auto training to obtain an appropriate model version.

Procedure

- On the **ExeML** page, click the name of the project that is successfully created. The **Label Data** tab page is displayed.

Figure 2-26 Finding unlabeled images



- On the **Label Data** tab page, click **Train** in the upper right corner. In the displayed **Training Configuration** dialog box, set related parameters. [Table 2-9](#) describes the parameters. Ensure that the number of decimal places of training and validation ratios ranges from 1 to 5.

Figure 2-27 Setting training parameters

Training Configuration

* Dataset Version

Training and Validation Ratios ? Training Set Ratio: ?
 Validation Set Ratio: 0.2

Max Training Time (Minute)

Training Preference ? ▼

Instance Flavor ▼

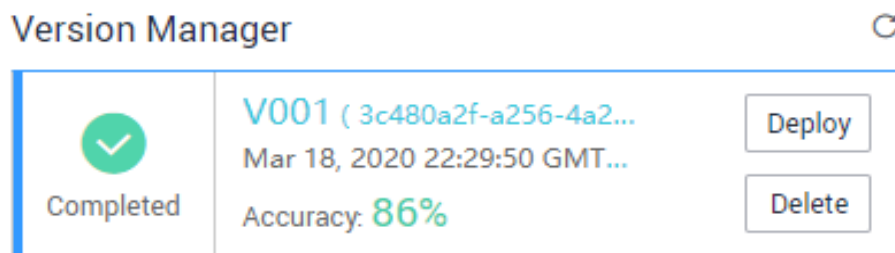
Table 2-9 Parameter description

Parameter	Description	Default Value
Dataset Version	This version is the one when the dataset is published in Data Management . In an ExeML project, when a training job is started, the dataset is published as a version based on the previous data labeling. The system automatically provides a version number. You can change it to the version number that you want.	Randomly provided by the system
Training and Validation Ratios	The labeled sample is randomly divided into a training set and a validation set. By default, the ratio for the training set is 0.8, and that for the validation set is 0.2. The usage field in the manifest file records the set type. The value ranges from 0 to 1.	0.8

Parameter	Description	Default Value
Incremental Training Version	Select the version with the highest precision to perform training again. This accelerates model convergence and improves training precision.	None
Max. Training Duration (Minute)	If training is not completed within the maximum training duration, the model is saved and training stops. To prevent the model from exiting before convergence, set this parameter to a large value. The value ranges from 6 to 6000. You are advised to properly extend the training duration. Set the training duration to more than 1 hour for a training set with 2,000 images.	60
Training Preference	<ul style="list-style-type: none"> ● performance_first: performance first. The training duration is short and the generated model is small. ● balance: balanced performance and precision ● accuracy_first: precision first. The training duration is long and the generated model is large. 	balance
Instance Flavor	<p>Select the resource specifications used for training. By default, the following specifications are supported:</p> <ul style="list-style-type: none"> ● Compute-intensive 1 instance (GPU): This flavor is billed on a pay-per-use basis. <p>The compute flavors are for reference only. Obtain the flavors on the management console.</p>	Free (GPU)

3. After configuring training parameters, click **Next** to go to the configuration page, confirm the specifications, and click **Submit** to start auto model training. The training takes a certain period of time. Wait until the training is complete. If you close or exit this page, the system still performs the training operation.
To use the free flavor, read the message carefully and select **I have read and agree to the above**.
4. On the **Train Model** tab page, wait until the training status changes from **Running** to **Completed**.

Figure 2-28 Running successful



- View the training details, such as **Accuracy**, **Evaluation Result**, **Training Parameters**, and **Classification Statistics**. For details about the evaluation result parameters, see [Table 2-10](#).

Figure 2-29 Model training result

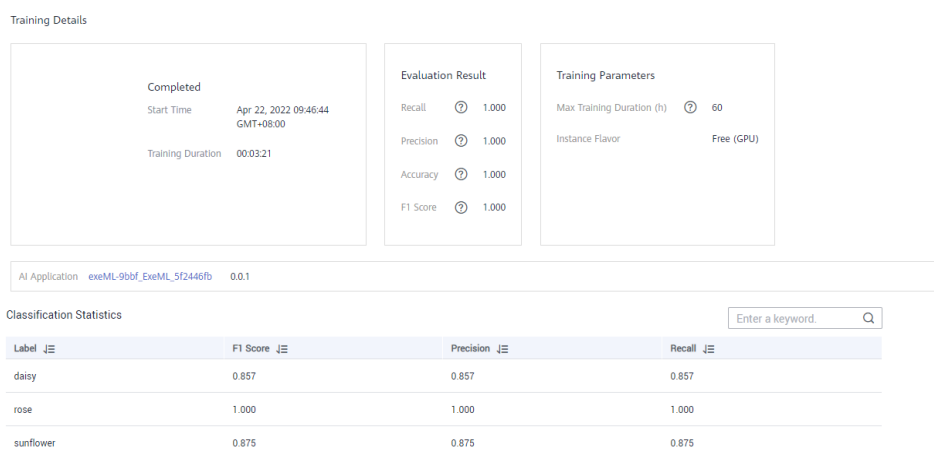


Table 2-10 Evaluation result parameters

Parameter	Description
Recall	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish positive samples.
Precision	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish negative samples.
Accuracy	Fraction of correctly predicted samples over all samples. It shows the general ability of a model to recognize samples.
F1 Score	Harmonic average of the precision and recall of a model. It is used to evaluate the quality of a model. A high F1 score indicates a good model.

NOTE

An ExeML project supports multiple rounds of training, and each round generates a version. For example, the first training version is **V001 (xxx)**, and the next version is **V002 (xxx)**. The trained models can be managed by training version. After the trained model meets your requirements, deploy the model as a service.

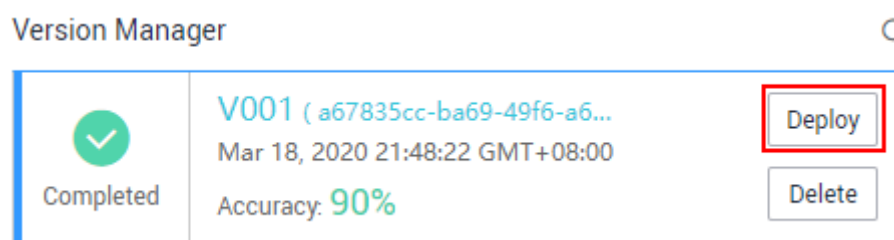
2.3.5 Deploying a Model as a Service

Procedure

You can deploy a model as a real-time service that provides a real-time test UI and monitoring capabilities. After the model is trained, you can deploy a **Completed** version with ideal accuracy as a service. The procedure is as follows:

1. On the **Train Model** tab page, wait until the training status changes to **Completed**. Click **Deploy** in the **Version Manager** pane to deploy the model as a real-time service.

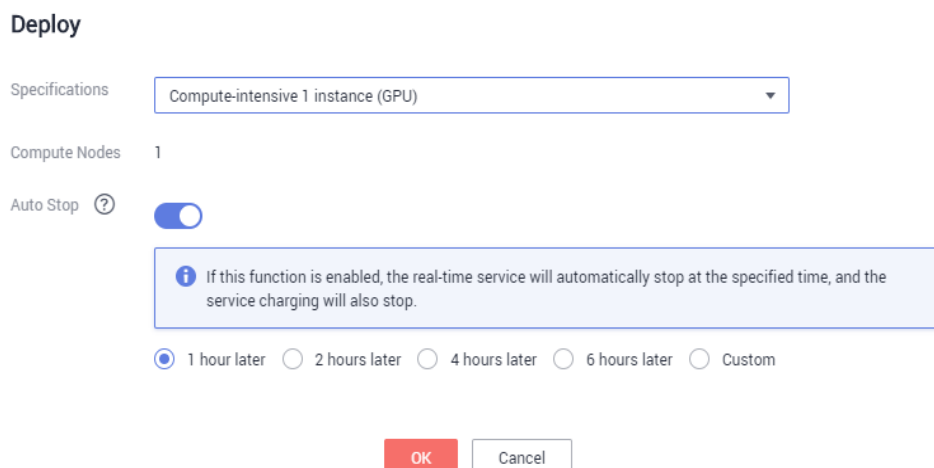
Figure 2-30 Deploy



2. In the **Deploy** dialog box, select resource flavor, set the **Auto Stop** function, and click **OK** to start the deployment.
 - **Specifications:** The GPU specifications are better, and the CPU specifications are more cost-effective.
 - **Compute Nodes:** The default value is **1** and cannot be changed.
 - **Auto Stop:** After this function is enabled and the auto stop time is set, a service automatically stops at the specified time. If this parameter is disabled, a real-time service keeps running and billing. The function can help you avoid unnecessary billing. The auto stop function is enabled by default, and the default value is **1 hour later**.

The options are **1 hour later**, **2 hours later**, **4 hours later**, **6 hours later**, and **Custom**. If you select **Custom**, enter any integer from 1 to 24 in the text box on the right.

Figure 2-31 Deploying a model



3. After the model deployment is started, view the deployment status on the **Service Deployment** page.

It takes a certain period of time to deploy a model. When the status in the **Version Manager** pane changes from **Deploying** to **Running**, the deployment is complete.

NOTE

On the **ExeML** page, trained models can only be deployed as real-time services. For details about how to deploy them as batch services or edge services, see [Where Are Models Generated by ExeML Stored? What Other Operations Are Supported?](#)

Testing a Service

- On the **Service Deployment** page, select a service type. For example, on the ExeML page, the object detection model is deployed as a real-time service by default. On the **Real-Time Services** page, click **Prediction** in the **Operation** column of the target service to perform a service test. For details, see [Testing the Deployed Service](#).
- You can also use code to test a service. For details, see [Accessing Real-Time Services](#).
- The following describes the procedure for performing a service test after the object detection model is deployed as a service on the ExeML page.
 - a. After the model is deployed, test the service using an image. On the **ExeML** page, click the target project, go to the **Deploy Service** tab page, select the service version in the **Running** status, click **Upload** in the service test area, and upload a local image to perform the test.

Figure 2-32 Uploading an image

Service Test

Prediction can be performed only when the service status is " Running ".

Select a file you want to use to test the service.



- b. Click **Predict** to perform the test. After the prediction is complete, the result is displayed in the **Test Result** pane on the right. If the model accuracy does not meet your expectation, add images on the **Label Data** tab page, label the images, and train and deploy the model again. **Table 2-11** describes the parameters in the prediction result. If you are satisfied with the model prediction result, call the API to access the real-time service as prompted. For details, see **Accessing Real-Time Services**. Currently, only JPG, JPEG, BMP, and PNG images are supported.

Figure 2-33 Prediction result

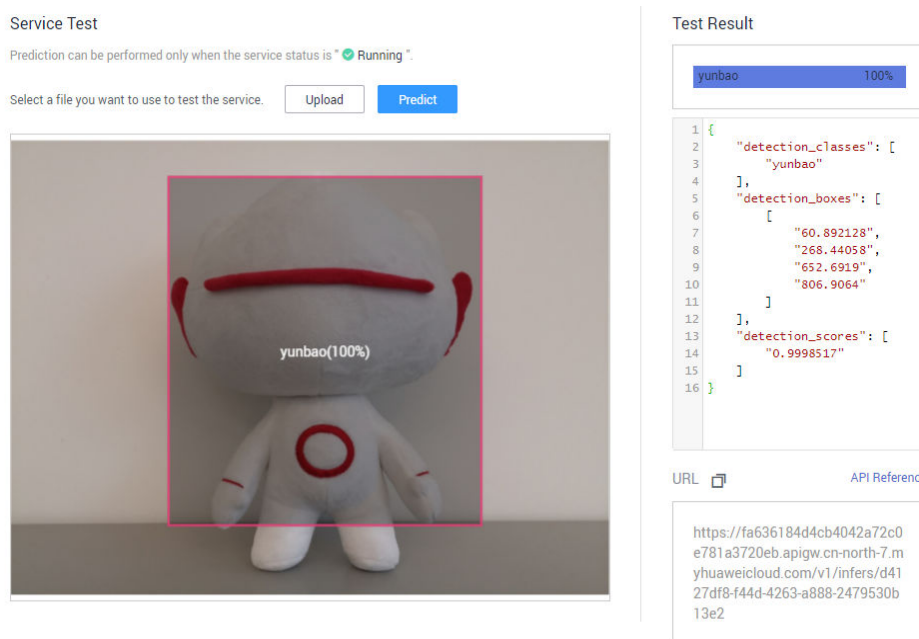
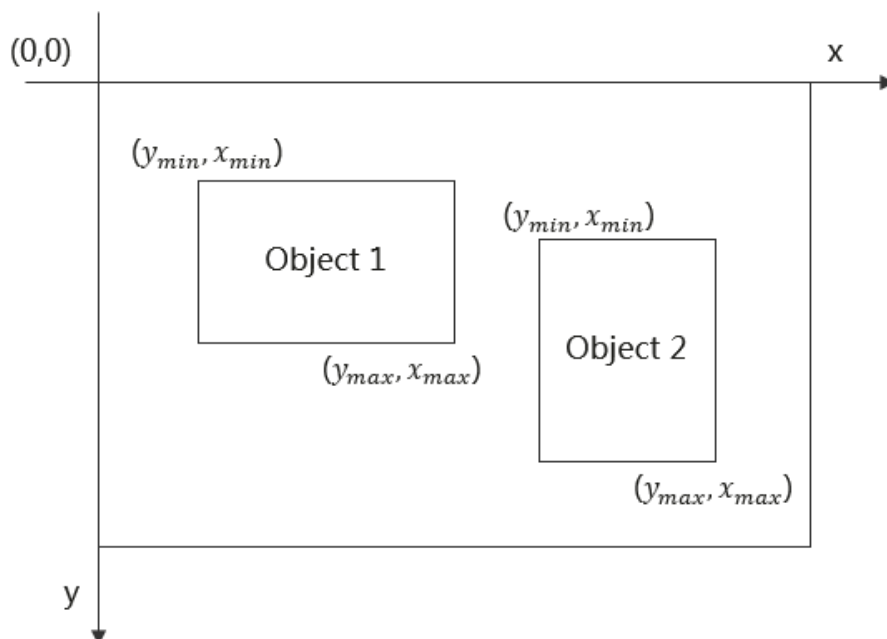


Table 2-11 Parameters in the prediction result

Parameter	Description
detection_classes	Label of each detection box
detection_boxes	Coordinates of four points (y_min, x_min, y_max, and x_max) of each detection box, as shown in Figure 2-34
detection_scores	Confidence of each detection box

Figure 2-34 Illustration for coordinates of four points of a detection box**NOTE**

A running real-time service keeps consuming resources. If you do not need to use the real-time service, click **Stop** in the **Version Manager** pane to stop the service so that charges will no longer be incurred. If you want to use the service again, click **Start**.

If you enable the auto stop function, the service automatically stops after the specified time and no fee is generated.

2.4 Predictive Analytics

2.4.1 Preparing Data

Before using ModelArts to build a predictive analytics model, upload data to OBS. The OBS bucket and ModelArts must be in the same region.

Uploading Data to OBS

This operation uses the OBS console to upload data.

Perform the following operations to import data to the dataset for model training and building.

1. Log in to OBS Console and **create a bucket** in the same region as ModelArts. If an available bucket exists, ensure that the OBS bucket and ModelArts are in the same region.
2. **Upload a file** to the OBS bucket. If you have a large amount of data, use OBS Browser+ to upload data or folders. The uploaded data must meet the dataset requirements of the ExeML project.

 NOTE

Upload data from unencrypted buckets. Otherwise, training will fail because data cannot be decrypted.

Requirements on Datasets

- The name of a file in a dataset consists of letters, digits, hyphens (-), and underscores (_), and the file name extension is CSV. The files cannot be stored in the root directory of an OBS bucket, but in a folder in the OBS bucket, for example, **/obs-xxx/data/input.csv**.
- The files are saved in CSV format. Use newline characters (\n) to separate lines and commas (,) to separate columns in the file. The file content cannot contain Chinese characters. The column content cannot contain special characters such as commas (,) and newline characters (\n). The quotation marks are not supported. It is recommended that the column content consist of letters and digits.
- Data training
 - The number of training columns is the same. There are at least 100 different data records in total (a feature with different values is considered as different data).
 - The training columns cannot contain timestamp data (such as yy-mm-dd or yyyy-mm-dd).
 - If a column has only one value, the column is considered invalid. Ensure that there are at least two values in the label column and no data is missing.

 NOTE

The label column is the training target specified in a training task. It is the output (prediction item) for the model trained using the dataset.

- In addition to the label column, the dataset must contain at least two valid feature columns. Ensure that there are at least two values in each feature column and that the percentage of missing data must be lower than 10%.
- The training data in CSV file cannot contain the table header. Otherwise, the training fails.
- Due to the limitation of the feature filtering algorithm, place the label column in the last column of the dataset. Otherwise, the training may fail.

Requirements for Files Uploaded to OBS

The OBS path of the predictive analytics projects must comply with the following rules:

- The OBS path of the input data must redirect to the data files. The data files must be stored in a folder in an OBS bucket rather than the root directory of the OBS bucket, for example, **/obs-xxx/data/input.csv**.
- The input data must be in CSV format. The data files do not contain the table header and the number of valid data lines must be greater than 100. The number of columns must be less than 200, and the total data size cannot exceed 100 MB.

Predictive Analytics File Example

Take the iris dataset as an example. Predict an iris species based on the lengths and widths of the iris calyx and petal.

Table 2-12 Parameters and meanings of data sources

Parameter	Meaning	Type	Description
attr_1	Calyx length	Double	Length of the target iris calyx
attr_2	Calyx width	Double	Width of the target calyx
attr_3	Petal length	Double	Length of the target iris petal
attr_4	Petal width	Double	Width of the target iris petal
attr_5	Species	String	Species of the iris

Table 2-13 Sample data

attr_1	attr_2	attr_3	attr_4	attr_5
5.1	3.5	1.4	0.2	Iris-setosa
7	3.2	4.7	1.4	Iris-versicolor
6.3	3.3	6	2.5	Iris-virginica

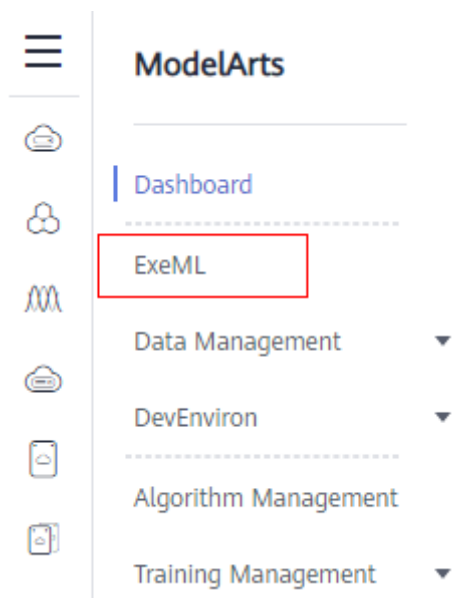
2.4.2 Creating a Project

ModelArts ExeML supports sound classification, text classification, image classification, predictive analytics, and object detection projects. You can create any of them based on your needs. Perform the following operations to create an ExeML project.

Procedure

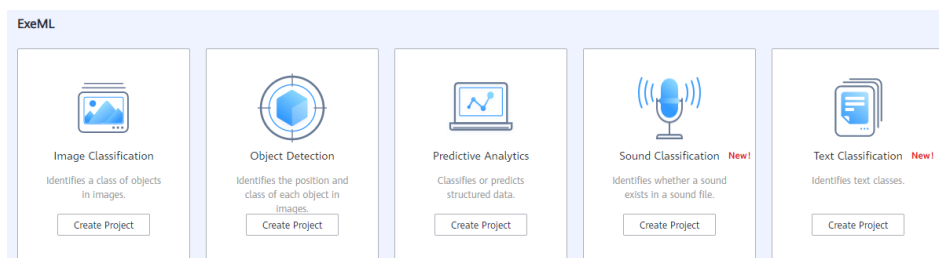
1. Log in to the ModelArts management console. In the left navigation pane, choose **ExeML**.

Figure 2-35 ExeML



2. Click **Create Project** in the box of your desired project. The page for creating an ExeML project is displayed.

Figure 2-36 ExeML project list



3. **Billing Mode** is default to **Pay-per-use**. Enter a project name and set **Training Data** to the OBS path of the training data. A data file must be specified in the path.

Table 2-14 Parameters

Parameter	Description
Name	Name of an ExeML project <ul style="list-style-type: none"> • Enter a maximum of 20 characters. Only digits, letters, underscores (_), and hyphens (-) are allowed. This parameter is mandatory. • The name must start with a letter.

Parameter	Description
Training Data	<p>OBS data path and data file. The selected OBS data path must meet certain specifications. For details, see Requirements for Files Uploaded to OBS.</p> <ul style="list-style-type: none"> Only the files and folders described in Preparing Data > Requirements for Files Uploaded to OBS can be saved in the training data path. Otherwise, an error will be reported. Do not modify the files in the training data path. <p>NOTE Only one ExeML project can be created in each OBS bucket path to training data. If you want to create multiple projects with the same dataset, copy the data in the original OBS bucket path to another OBS bucket path, and then create an ExeML project.</p>
Description	Brief description of a project

- Click **Create Project**. The system displays a message indicating that the project has been created. Then, the **Label Data** tab page is displayed. You can also view the project whose **Training Status** is **Not Started** on the **ExeML** page. Click the project name to go to the **Label Data** tab page.

2.4.3 Selecting a Label Column

After creating a predictive analytics project, select a label column and its data type. On the **Label Data** tab page, you can preview data and select the label column and its data type. Due to the limitation of the feature filtering algorithm, the label column must be the last column of the dataset. During model training, all data is used to train an inference model. The model uses the data of other columns as the input and outputs the inference value in the label column.

Procedure

- Select a label column. On the **Label Data** tab page, preview the data and select the training objective. Select the label column from the drop-down list of **Label Column**.
The label column is the output of an inference model. In this project, the training objective is to identify iris species. The inference data will be output as discrete values in column **attr_5**. After the training objective is specified, click **Train**.
- Select the data type of the label column. On the **Label Data** tab page, select a data type for **Label Column Data Type**.
 - If the label column contains enumeration data, select **Discrete value**. The predictive analytics project will train a classification model.
 - If the label column contains continuous numeric data, select **Continuous value**. The predictive analytics project will train a regression model.

NOTE

- For discrete values, the evaluation result shows the recall, precision, accuracy, and F1 score after the model training is complete.
- For continuous values, the evaluation result shows mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE) after model training is complete.

2.4.4 Training a Model

After the data is labeled, train a model for predictive analytics. You can publish the model as a real-time inference service.

Procedure

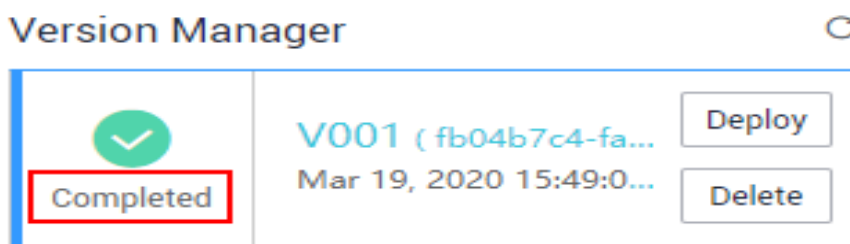
1. On the **ExeML** page, click the name of the project that has been created. The **Label Data** tab page is displayed. Select the label column and its data type.
2. On the **Label Data** tab page, click **Train** in the lower left corner. In the displayed **Training Configuration** dialog box, select an instance flavor used for training, click **Next** to go to the configuration page, confirm the specifications, and click **Submit** to start model training.

For ExeML projects of predictive analytics, only the **ExeML CPU (8U) instance** flavor can be used for model training.

The training takes a certain period of time. If you close or exit the page, the system continues training until it is complete.

3. On the **Train Model** tab page, wait until the training status changes from **Running** to **Completed**.

Figure 2-37 Successful running



4. View the training details, such as the label column, data type, accuracy, and evaluation result.

The example is a discrete value of binary classification. For details about the evaluation result parameters, see [Table 2-15](#).

For details about the evaluation results generated for different data types of label columns, see [Evaluation Results](#).

NOTE

An ExeML project supports multiple rounds of training, and each round generates a version. For example, the first training version is **V001 (xxx)**, and the next version is **V002 (xxx)**. The trained models can be managed by training version. After the trained model meets your requirements, deploy the model as a service.

Evaluation Results

The parameters in evaluation results vary depending on the training data type.

- Discrete values

The evaluation parameters include recall, precision, accuracy, and F1 score, which are described in the following table.

Table 2-15 Parameters in discrete value evaluation results

Parameter	Description
Recall	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish positive samples.
Precision	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish negative samples.
Accuracy	Fraction of correctly predicted samples over all samples. It shows the general ability of a model to recognize samples.
F1 Score	Harmonic average of the precision and recall of a model. It is used to evaluate the quality of a model. A high F1 score indicates a good model.

- Continuous values

The evaluation parameters include Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). The three error values represent a difference between a real value and a predicted value. During multiple rounds of modeling, a group of error values is generated for each round of modeling. Use these error values to determine the quality of a model. A smaller error value indicates a better model.

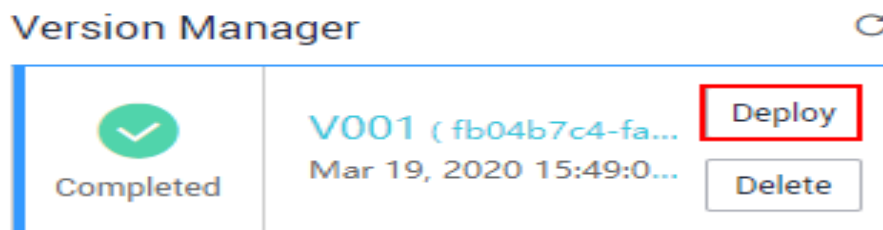
2.4.5 Deploying a Model as a Service

Deploying a Model

You can deploy a model as a real-time service that provides a real-time test UI and monitoring capabilities. After the model is trained, you can deploy a **Successful** version with ideal accuracy as a service. The procedure is as follows:

1. On the **Train Model** tab page, wait until the training status changes to **Completed**. Click **Deploy** in the **Version Manager** pane.

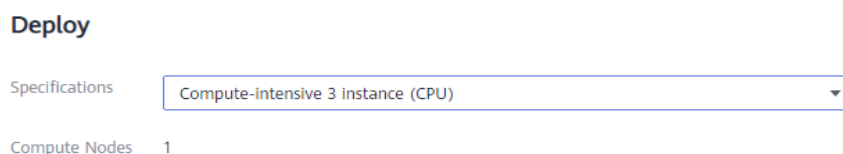
Figure 2-38 Deploy



2. In the displayed **Deploy** dialog box, set **Specifications**, and click **OK** to deploy the model as a real-time service.
 - **Specifications:**
 - **Compute Nodes:** The default value is **1** and cannot be changed.
 - **Auto Stop:** After this function is enabled and the auto stop time is set, a service automatically stops at the specified time. If this function is disabled, a real-time service will continue to run and charges will continue to be incurred. The auto stop function is enabled by default. The default value is **1 hour later**.

The options are **1 hour later**, **2 hours later**, **4 hours later**, **6 hours later**, and **Custom**. If you select **Custom**, enter any integer from 1 to 24 in the text box on the right.

Figure 2-39 Deploying a model



3. After the model deployment is started, view the deployment status on the **Service Deployment** page.

It takes a certain period of time to deploy a model. When the status in the **Version Manager** pane changes from **Deploying** to **Running**, the deployment is complete.

NOTE

On the **ExeML** page, trained models can only be deployed as real-time services. For details about how to deploy them as batch services or edge services, see [Where Are Models Generated by ExeML Stored? What Other Operations Are Supported?](#)

Testing the Service

- On the **Service Deployment** page, select a service type. For example, the predictive analytics model on the **ExeML** page is deployed as a real-time service by default. Then, select **Real-Time Services** in the left pane and click **Predict** in the **Operation** column of the target service to test the service. For details, see [Testing the Deployed Service](#).
- You can also use code to test a service. For details, see [Accessing Real-Time Services](#).

- The following describes the procedure for performing a service test after the predictive analytics model is deployed as a service on the ExeML page.
 - a. After the model is deployed, you can test the model using code. On the **ExeML** page, click the target project, go to the **Deploy Service** tab page, select a service version in the **Running** state, and enter the code in the **Service Test** pane.
 - b. Click **Predict** to perform the test. After the prediction is complete, the result is displayed in the **Test Result** pane on the right. If the model accuracy does not meet your expectation, train and deploy the model again on the **Label Data** tab page. If you are satisfied with the model prediction result, call the API to access the real-time service as prompted. For details, see [Accessing Real-Time Services](#).

- **attr_1** to **attr_4** indicate the input data. On the **Label Data** tab page, the selected label column is **attr_5**, indicating that **attr_5** is the target column to be predicted.

```
{
  "data": {
    "req_data": [{
      "attr_1": 5.1,
      "attr_2": 3.5,
      "attr_3": 1.4,
      "attr_4": 0.2
    }]
  }
}
```

- In the preceding code snippet, **predict** is the inference result of label column **attr_5**. See [Figure 2-40](#).

Figure 2-40 Prediction result

Code	Return Result
<pre>1 { 2 "data": { 3 "req_data": [{ 4 "attr_1": 5.1, 5 "attr_2": 3.5, 6 "attr_3": 1.4, 7 "attr_4": 0.2 8 }] 9 } 10 } 11 }</pre>	<pre>1 { 2 "data": { 3 "resp_data": [4 { 5 "predict": "Iris-setosa" 6 } 7] 8 } 9 }</pre>

NOTE

A running real-time service keeps consuming resources. If you do not need to use the real-time service, click **Stop** in the **Version Manager** pane to stop the service so that charges will no longer be incurred. If you want to use the service again, click **Start**.

2.5 Sound Classification

2.5.1 Preparing Data

Before using ModelArts ExeML to build a model, upload data to an OBS bucket. The OBS bucket and ModelArts must be in the same region.

Uploading Data to OBS

This operation uses the OBS client to upload data. For more information about how to create a bucket and upload files, see [Creating a Bucket](#) and [Uploading an Object](#).

Perform the following operations to import data to the dataset for model training and building.

1. Log in to OBS Console and [create a bucket](#) in the same region as ModelArts. If an existing bucket is available, ensure that the OBS bucket and ModelArts are in the same region.
2. [Upload a file](#) to the OBS bucket. If you have a large amount of data, use OBS Browser+ to upload data or folders. The uploaded data must meet the dataset requirements of the ExeML project.

NOTE

Upload data from unencrypted buckets. Otherwise, training will fail because data cannot be decrypted.

Requirements for Sound Classification Data

- Only 16-bit WAV files are supported. All sub-formats of WAV are supported.
- The duration of a sound file must be longer than 1 second, and the maximum size of a sound file is 4 MB.
- Add more sound files to a training set, improving model precision. Prepare at least 50 sound files for each class, and the total length of each class of sound files must be at least 5 minutes.
- Ensure that the sound files are authentic, and that each class of sound files covers all application scenarios in the real world.
- The quality of the training set has a great impact on the precision of the model. It is recommended that the sampling rate and precision of the training set be the same.
- The labeling quality has a great impact on the model precision. Do not mislabel objects.
- Only Chinese and English are supported for audio labeling.

Requirements for Files Uploaded to OBS

- If you do not need to upload training data in advance, create an empty folder to store files generated in the future, for example, **/bucketName/data-cat**.
- If you need to upload sound files to be labeled in advance, create an empty folder and save the sound files in the folder. An example of the file directory structure is **/bucketName/data-cat/cat.wav**.

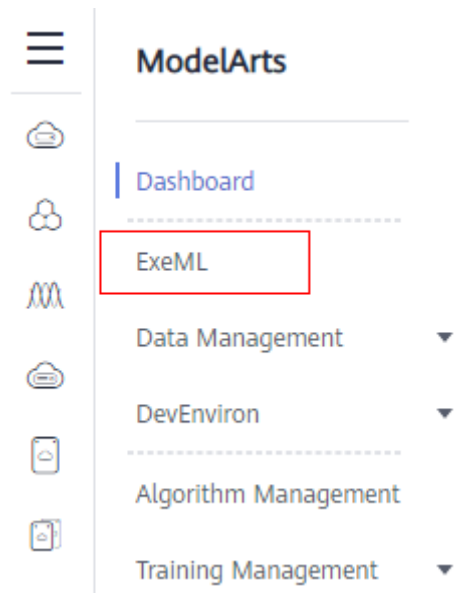
2.5.2 Creating a Project

ModelArts ExeML supports sound classification, text classification, image classification, predictive analytics, and object detection projects. You can create any of them based on your needs. Perform the following operations to create an ExeML project.

Procedure

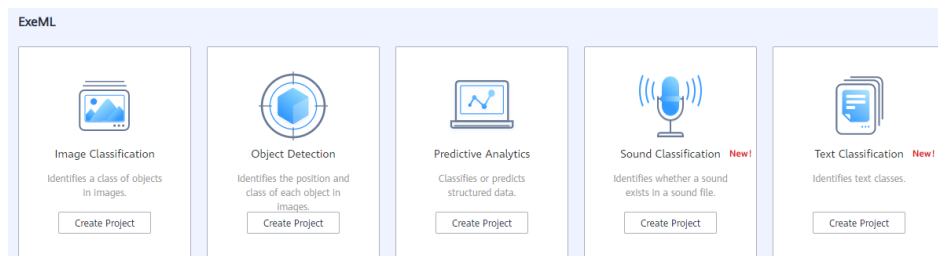
1. Log in to the ModelArts management console. In the left navigation pane, choose **ExeML**.

Figure 2-41 ExeML



2. Click **Create Project** in the box of your desired project. The page for creating an ExeML project is displayed.

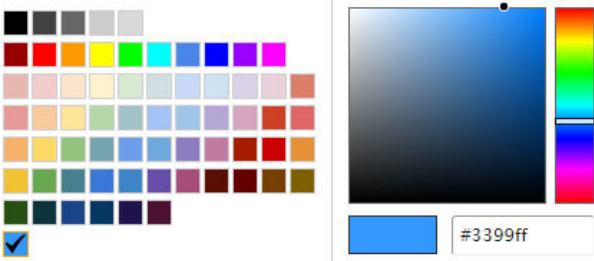
Figure 2-42 ExeML project list



3. On the displayed page, set the parameters by referring to [Table 2-16](#). The default billing mode is **Pay-per-use**.

Table 2-16 Parameters

Parameter	Description
Name	<p>Name of an ExeML project</p> <ul style="list-style-type: none"> • Enter a maximum of 32 characters. Only digits, letters, underscores (_), and hyphens (-) are allowed. This parameter is mandatory. • The name must start with a letter.
Description	Brief description of a project

Parameter	Description
Dataset Source	<p>You can create a dataset or specify an existing dataset.</p> <ul style="list-style-type: none"> • Create: Configure parameters such as Dataset Name, Input Dataset Path, Output Dataset Path, and Label Set. • Specify: Select a dataset of the same type from ModelArts Data Management to create an ExeML project. Only datasets of the same type are displayed in the Dataset Name drop-down list.
Dataset Name	<p>If you select Create for Dataset Source, enter a dataset name based on required rules in the text box on the right. If you select Specify for Dataset Source, select one from available datasets of the same type under the current account displayed in the drop-down list.</p>
Input Dataset Path	<p>Select the OBS path to the input dataset. For details about dataset input specifications, see Preparing Data.</p> <ul style="list-style-type: none"> • Only the files and folders described in Preparing Data > Requirements for Files Uploaded to OBS can be saved in the training data path. Otherwise, an error will be reported. • Do not modify the files in the training data path.
Output Dataset Path	<p>Select the OBS path for storing the output dataset.</p> <p>NOTE The output dataset path cannot be the same as the input dataset path or cannot be the subdirectory of the input dataset path. It is a good practice to select an empty directory in Output Dataset Path.</p>
Label Set	<ul style="list-style-type: none"> • Label Name: Enter a label name. The label name can contain only Chinese characters, letters, digits, underscores (_), and hyphens (-), which contains 1 to 32 characters. • Add Label: Click Add Label to add one or more labels. • Set the label color: You need to set label colors for object detection and text classification datasets, but you do not need to set label colors for image and sound classification datasets. Select a color from the color palette on the right of a label, or enter the hexadecimal color code to set the color. 

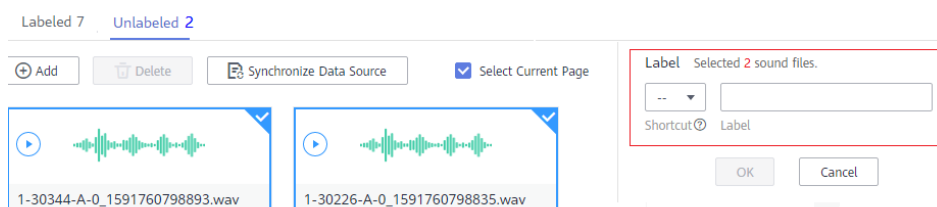
- Click **Create Project**. The system displays a message indicating that the project has been created. Then, the **Label Data** tab page is displayed. Alternatively, view the created project on the **ExeML** page and click the project name to go to the **Label Data** page.

2.5.3 Labeling Data

Labeling Sound Files

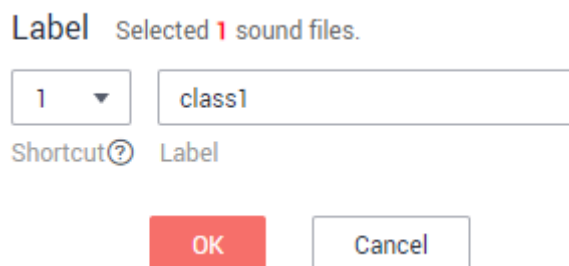
- Select an unlabeled sound file. On the **Label Data** tab page, click the **Unlabeled** tab. All unlabeled sound files are displayed. Select the sound files to be labeled in sequence, or tick **Select Current Page** to select all sound files on the page, and then add labels to the sound files in the right pane.

Figure 2-43 Labeling a sound file



- Add a label. Play a sound file, and select the sound file. In the **Label** area, enter a label name or select an existing label from the drop-down list on the right, and select a shortcut from the drop-down list on the left. Click **OK**. The selected sound file is labeled.

Figure 2-44 Labeling a sound file



- After all the sound files are labeled, view them on the **Labeled** tab page or view the list of **All Labels** in the right pane to learn the name and quantity of the labels.

Synchronizing or Adding Sound Files

On the **ExeML** page, click the project name. The **Label Data** tab page is displayed. When creating a sound classification project, you can select local data or synchronize data in OBS as the training data.

- Add Audio:** You can quickly add sound files on a local PC to ModelArts and synchronize the files to the OBS path specified during project creation. Click **Add Audio**. In the dialog box that is displayed, click **Add Audio** and add sound files.

 **NOTE**

Only 16-bit WAV files are supported. The size of a sound file cannot exceed 4 MB. The total size of all sound files uploaded in one attempt cannot exceed 8 MB.

- **Synchronize Data Source:** To quickly obtain the latest sound files in the OBS bucket, click **Synchronize Data Source** to add sound files in OBS to ModelArts.
- **Delete Audio:** You can delete sound files one by one, or tick **Select Current Page** to delete all sound files on the page.

 **NOTE**

The deleted sound files cannot be recovered. Exercise caution when performing this operation.

Modifying Labeled Data

After labeling data, you can modify the labeled data on the **Labeled** tab page.

- **Modifying based on audio**

On the dataset details page, click the **Labeled** tab. Select one or more audio files to be modified from the audio list. Modify the label in the label details area on the right.






- **Modifying a label:** In the **File Labels** area, click the editing icon in the **Operation** column, enter the correct label name in the text box, and click the check mark icon.
- **Deleting a label:** In the **File Labels** area, click the deletion icon in the **Operation** column. In the displayed dialog box, click **OK**.

- **Modifying based on labels**

On the **Label Data** tab page, click the **Labeled** tab. The information about all labels is displayed on the right.

Figure 2-45 Information about all labels

All Labels 2

Label	Count 	Shortcut	Operation
class2	5	2	 
class1	5	1	 

- **Modifying a label:** Click the editing icon in the **Operation** column. In the dialog box that is displayed, enter the new label name, select the new shortcut key, and click **OK**. After the modification, the new label applies to the audio files that contain the original label.
- **Deleting a label:** Click the deletion icon in the **Operation** column. In the displayed dialog box, select the object to be deleted as prompted and click **OK**.

2.5.4 Training a Model

After labeling the sound files, train a model. You can perform model training to obtain the required sound classification model. Training sound files must be classified into at least two classes, and each class must contain at least five sound files. Before training, ensure that the labeled sound files meet the requirements. Otherwise, the **Train** button is unavailable.

Procedure

Before starting the training, set the training parameters and then perform auto training.

1. On the **ExeML** page, click the name of the project that is successfully created. The **Label Data** tab page is displayed.

Figure 2-46 Finding unlabeled files



2. On the **Label Data** tab page, click **Train** in the upper right corner. In the displayed **Training Configuration** dialog box, set related parameters by referring to [Table 2-17](#) and click **OK** to start model training.

Figure 2-47 Setting training parameters

Training Configuration

* Dataset Version	<input type="text" value="V016"/>
Training and Validation Ratios ?	Training Set Ratio: <input type="text" value="0.8"/> ? Validation Set Ratio: 0.2
Max Training Time (Minute)	<input type="text" value="60"/>
Training Preference ?	<input type="text" value="balance"/>
Instance Flavor	<input type="text" value="Free (GPU)"/>

Table 2-17 Parameter description

Parameter	Description	Default Value
Dataset Version	<p>This version is the one when the dataset is published in Data Management. In an ExeML project, when a training job is started, the dataset is published as a version based on the previous data labeling.</p> <p>The system automatically provides a version number. You can change it to the version number that you want.</p>	Randomly provided by the system
Max. Training Duration (Minute)	<p>If the training is not completed within the maximum training duration, the training is forcibly stopped. You are advised to enter a larger value to prevent forcible stop during training. The value ranges from 6 to 6000.</p>	60
Instance Flavor	<p>Select the resource specifications used for training. By default, the following types are supported:</p> <ul style="list-style-type: none"> • Compute-intensive 1 instance (GPU): This flavor is billed on a pay-per-use basis. • Free (GPU): This flavor is free. However, if the flavor is used, the training job automatically stops after one hour. That is, the training job lasts for only one hour at a time. You are advised to evaluate the data size and ensure that the time of a training job does not exceed 1 hour. When there are a large number of users, they need to wait in a queue for this flavor. 	ExeML (GPU)

3. After configuring training parameters, click **Next** to go to the configuration page, confirm the specifications, and click **Submit** to start auto model training. The training takes a certain period of time. Wait until the training is complete. If you close or exit this page, the system still performs the training operation.
4. On the **Train Model** tab page, wait until the training status changes from **Running** to **Completed**.
5. View the training details, such as **Accuracy**, **Evaluation Result**, **Training Parameters**, and **Classification Statistics**.

Figure 2-48 Training details

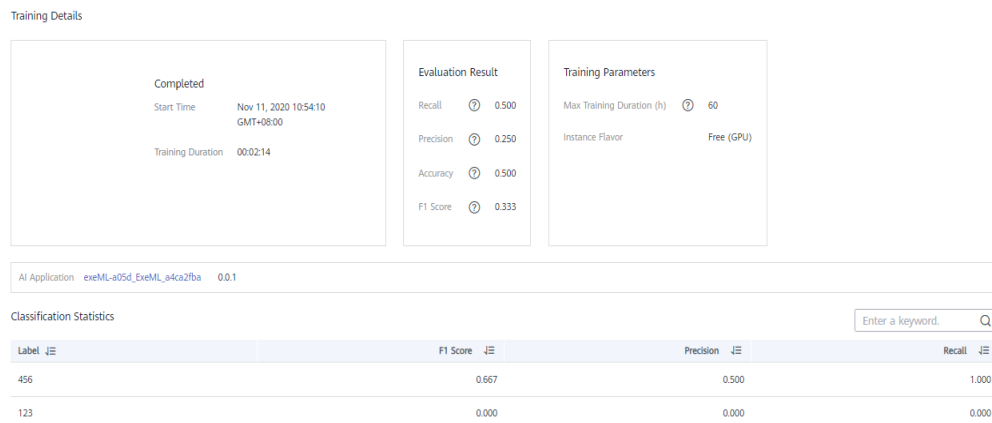


Table 2-18 Evaluation result parameters

Parameter	Description
Recall	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish positive samples.
Precision	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish negative samples.
Accuracy	Fraction of correctly predicted samples over all samples. It shows the general ability of a model to recognize samples.
F1 Score	Harmonic average of the precision and recall of a model. It is used to evaluate the quality of a model. A high F1 score indicates a good model.

NOTE

An ExeML project supports multiple rounds of training, and each round generates a version. For example, the first training version is **V001 (xxx)**, and the next version is **V002 (xxx)**. The trained models can be managed by training version. After the trained model meets your requirements, deploy the model as a service.

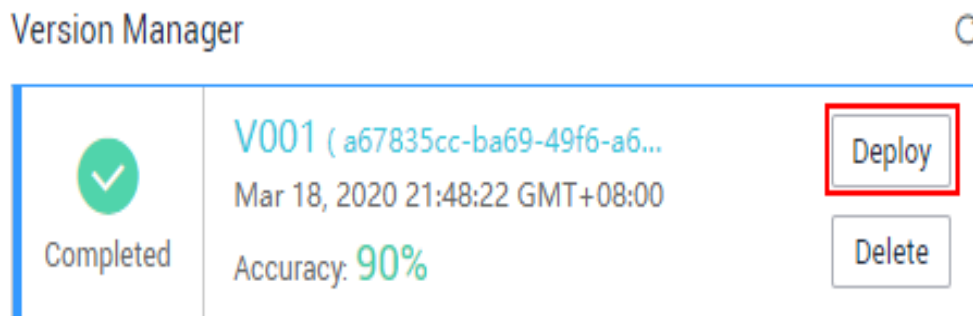
2.5.5 Deploying a Model as a Service

Procedure

You can deploy a model as a real-time service that provides a real-time test UI and monitoring capabilities. After the model is trained, you can deploy a **Completed** version with ideal accuracy as a service. The procedure is as follows:

1. On the **Train Model** tab page, wait until the training status changes to **Completed**. Click **Deploy** in the **Version Manager** pane to deploy the model as a real-time service.

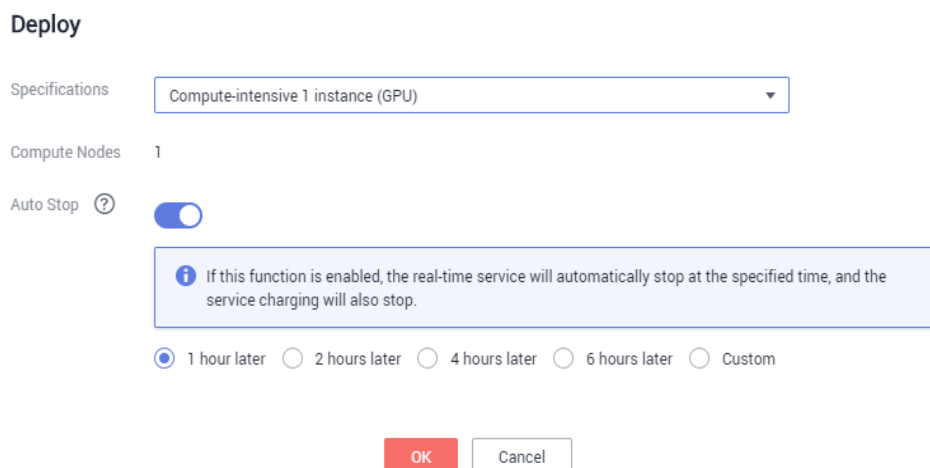
Figure 2-49 Deploy



2. In the **Deploy** dialog box, select resource flavor, set the **Auto Stop** function, and click **OK** to start the deployment.
 - **Specifications:** The GPU specifications are better, and the CPU specifications are more cost-effective.
 - **Compute Nodes:** The default value is **1** and cannot be changed.
 - **Auto Stop:** After this function is enabled and the auto stop time is set, a service automatically stops at the specified time. If this parameter is disabled, a real-time service keeps running and billing. The function can help you avoid unnecessary billing. The auto stop function is enabled by default, and the default value is **1 hour later**.

The options are **1 hour later**, **2 hours later**, **4 hours later**, **6 hours later**, and **Custom**. If you select **Custom**, enter any integer from 1 to 24 in the text box on the right.

Figure 2-50 Deploying a model



3. After the model deployment is started, view the deployment status on the **Service Deployment** page.
It takes a certain period of time to deploy a model. When the status in the **Version Manager** pane changes from **Deploying** to **Running**, the deployment is complete.

 NOTE

On the **ExeML** page, trained models can only be deployed as real-time services. For details about how to deploy them as batch services or edge services, see [Where Are Models Generated by ExeML Stored? What Other Operations Are Supported?](#)

Testing a Service

- On the **Service Deployment** page, select a service type. For example, on the ExeML page, the sound classification model is deployed as a real-time service by default. On the **Real-Time Services** page, click **Prediction** in the **Operation** column of the target service to perform a service test. For details, see [Testing the Deployed Service](#).
- You can also use code to test a service. For details, see [Accessing Real-Time Services](#).
- The following describes the procedure for performing a service test after the sound classification model is deployed as a service on the ExeML page.
 - a. After the model is deployed, you can add a sound file for test. On the **ExeML** page, click the target project, go to the **Deploy Service** tab page, select the service version in the **Running** status, click **Upload** in the service test area, and upload a local sound file to perform the test.
 - b. Click **Predict** to perform the test. After the prediction is complete, the result is displayed in the **Test Result** pane on the right. If the model accuracy does not meet your expectation, add sound files on the **Label Data** tab page, label the files, and train and deploy the model again. [Table 2-19](#) describes the parameters in the prediction result. If you are satisfied with the model prediction result, call the API to access the real-time service as prompted. For details, see [Accessing Real-Time Services](#).

Table 2-19 Parameters in the prediction result

Parameter	Description
predicted_label	Prediction type of the audio segment
score	Confidence score for the predicated class

 NOTE

A running real-time service keeps consuming resources. If you do not need to use the real-time service, click **Stop** in the **Version Manager** pane to stop the service so that charges will no longer be incurred. If you want to use the service again, click **Start**.

If you enable the auto stop function, the service automatically stops after the specified time and no fee is generated.

2.6 Text Classification

2.6.1 Preparing Data

Before using ModelArts ExeML to build a model, upload data to an OBS bucket. The OBS bucket and ModelArts must be in the same region.

Uploading Data to OBS

There are many restrictions on using the OBS Console, and the OBS client is used to upload data. For more information about how to create a bucket and upload files, see [Creating a Bucket](#) and [Uploading an Object](#).

Perform the following operations to import data to the dataset for model training and building.

1. Log in to OBS Console and [create a bucket](#) in the same region as ModelArts. If an available bucket exists, ensure that the OBS bucket and ModelArts are in the same region.
2. [Upload the local data](#) to the OBS bucket. If you have a large amount of data, use OBS Browser+ to upload data or folders. The uploaded data must meet the dataset requirements of the ExeML project.

NOTE

Upload data from unencrypted buckets. Otherwise, training will fail because data cannot be decrypted.

Requirements on Datasets

- Files must be in TXT or CSV format, and cannot exceed 8 MB.
- Use line feed characters to separate rows in files, and each row of data represents a labeled object.
- Currently, text classification supports only Chinese.

Requirements for Files Uploaded to OBS

- If you do not need to upload training data in advance, create an empty folder to store files generated in the future.
- If you need to upload files to be labeled in advance, create an empty folder and save the files in the folder. An example of the file directory structure is / **bucketName/data/text.csv**.
- A label name can contain a maximum of 32 characters, including letters, digits, hyphens (-), and underscores (_).
- If you want to upload labeled text files to the OBS bucket, upload them according to the following specifications:
 - The objects and files to be labels must be in the same directory. The objects must be in one-to-one relationship with the files. For example, if the object file name is **COMMENTS_114745.txt**, the label file name must be **COMMENTS_114745_result.txt**.

Example of data files:

```
|<dataset-import-path>
|  COMMENTS_114732.txt
|  COMMENTS_114732_result.txt
|  COMMENTS_114745.txt
```

```
COMMENTS_114745_result.txt
COMMENTS_114945.txt
COMMENTS_114945_result.txt
```

- The labeled objects and label files for text classification are text files, and correspond to each other based on rows. For example, the first row in a label file indicates the label of the first row in the labeled object.

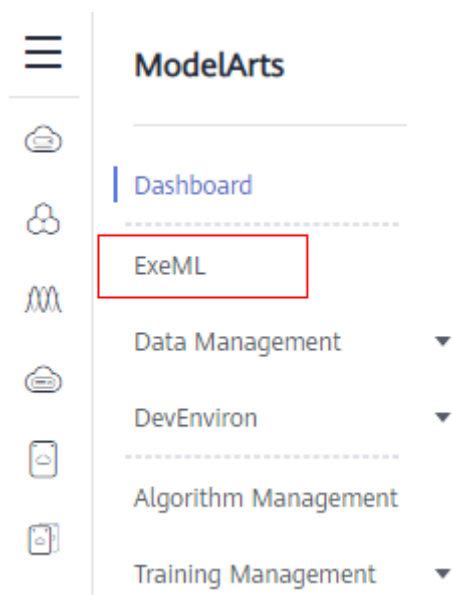
2.6.2 Creating a Project

ModelArts ExeML supports sound classification, text classification, image classification, predictive analytics, and object detection projects. You can create any of them based on your needs. Perform the following operations to create an ExeML project.

Procedure

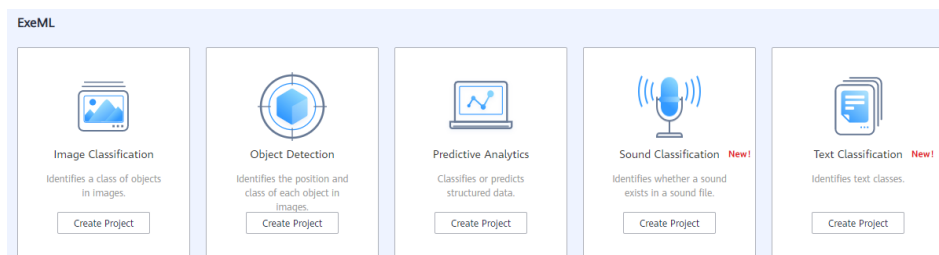
1. Log in to the ModelArts management console. In the left navigation pane, choose **ExeML**.

Figure 2-51 ExeML



2. Click **Create Project** in the box of your desired project. The page for creating an ExeML project is displayed.

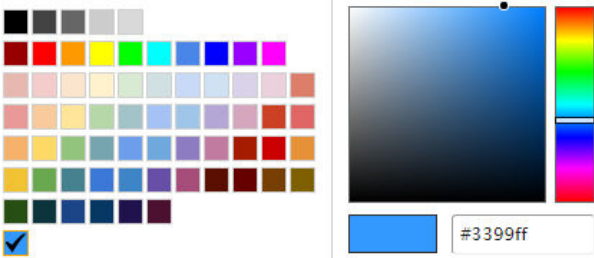
Figure 2-52 ExeML project list



3. On the displayed page, set the parameters by referring to [Table 2-20](#). The default billing mode is **Pay-per-use**.

Table 2-20 Parameters

Parameter	Description
Name	<p>Name of an ExeML project</p> <ul style="list-style-type: none"> Enter a maximum of 32 characters. Only digits, letters, underscores (_), and hyphens (-) are allowed. This parameter is mandatory. The name must start with a letter.
Description	Brief description of a project
Dataset Source	<p>You can create a dataset or specify an existing dataset.</p> <ul style="list-style-type: none"> Create: Configure parameters such as Dataset Name, Input Dataset Path, Output Dataset Path, and Label Set. Specify: Select a dataset of the same type from ModelArts Data Management to create an ExeML project. Only datasets of the same type are displayed in the Dataset Name drop-down list.
Dataset Name	<p>If you select Create for Dataset Source, enter a dataset name based on required rules in the text box on the right. If you select Specify for Dataset Source, select one from available datasets of the same type under the current account displayed in the drop-down list.</p>
Input Dataset Path	<p>Select the OBS path to the input dataset. For details about dataset input specifications, see Preparing Data.</p> <ul style="list-style-type: none"> Only the files and folders described in Preparing Data > Requirements for Files Uploaded to OBS can be saved in the training data path. Otherwise, an error will be reported. Do not modify the files in the training data path.
Output Dataset Path	<p>Select the OBS path for storing the output dataset.</p> <p>NOTE The output dataset path cannot be the same as the input dataset path or cannot be the subdirectory of the input dataset path. It is a good practice to select an empty directory in Output Dataset Path.</p>

Parameter	Description
Label Set	<ul style="list-style-type: none"> ● Label Name: Enter a label name. The label name can contain only Chinese characters, letters, digits, underscores (_), and hyphens (-), which contains 1 to 32 characters. ● Add Label: Click Add Label to add one or more labels. ● Set the label color: You need to set label colors for object detection and text classification datasets, but you do not need to set label colors for image and sound classification datasets. Select a color from the color palette on the right of a label, or enter the hexadecimal color code to set the color. 

4. Click **Create Project**. The system displays a message indicating that the project has been created. Then, the **Label Data** tab page is displayed. Alternatively, view the created project on the **ExeML** page and click the project name to go to the **Label Data** page.

2.6.3 Labeling Data

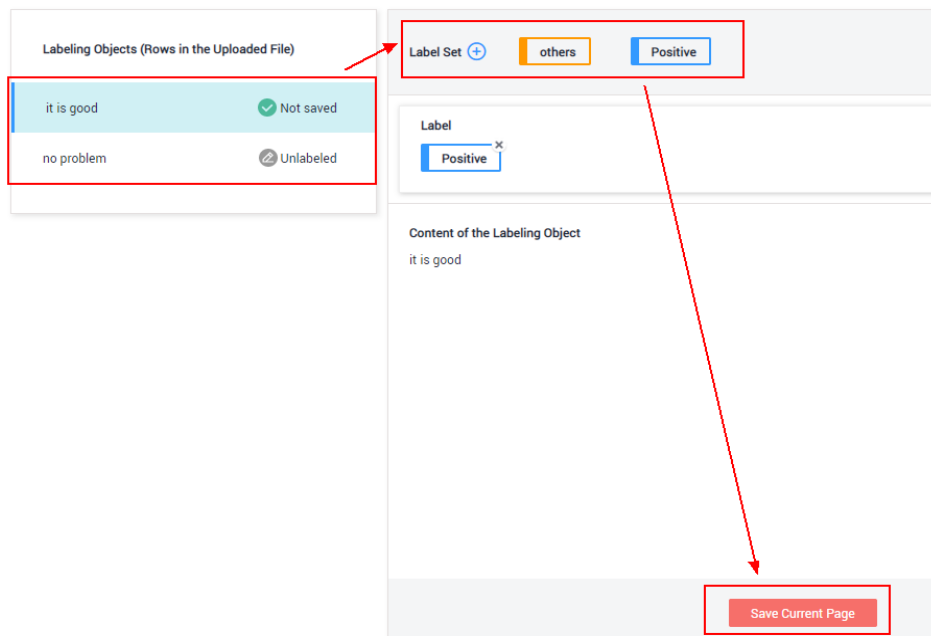
After a text classification project is created, the **ExeML > Label Data** tab page is displayed. The **Labeled** tab page is displayed by default. If the selected dataset contains labeled data, the labeled data is automatically displayed. You can also click **Unlabeled** to switch to the **Unlabeled** tab page. Unlabeled data in the directory of the target dataset is displayed.

Data Labeling for Text Classification

1. Select a text to be labeled in **Labeling Objects** and click different labels in the **Label Set** area to label the text.
You can add only one label for a text object.
2. After confirming the file label, click **Save Current Page** in the lower right corner to save the labeling.

If a large number of objects are included in **Labeling Objects**, the page turning icon is displayed in the lower part of the area. After labeling objects on this page, click **Save Current Page** before you turn to the next page. If you turn pages before saving the labellings, the labeling information on the previous page will be lost. You need to re-label for text data.

Figure 2-53 Data labeling - text classification

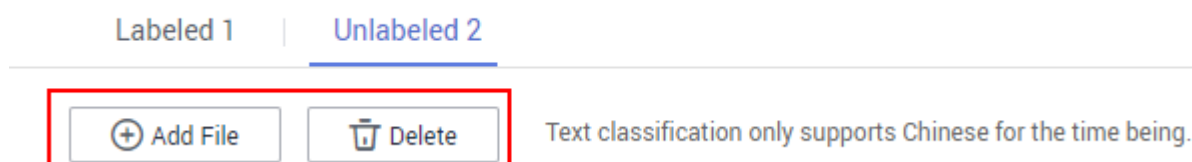


Adding or Deleting Data

In an ExeML project, the data source is the OBS directory corresponding to the input path of the dataset. If the data in the directory cannot meet your requirements, add or delete data on the ExeML page of ModelArts.

- Adding a file**
 On the **Unlabeled** tab page, click **Add File** in the upper left corner. In the dialog box that is displayed, select a local file and upload it.
 The format of the file to be uploaded must meet [requirement on datasets](#) of the text classification type.
- Deleting a text object**
 On the **Labeled** or **Unlabeled** tab page, select a text object to be deleted and click **Delete** in the upper left corner. In the dialog box that is displayed, confirm the deletion information and click **OK**.
 On the **Labeled** tab page, you can tick **Select Current Page** and click **Delete** to delete all text objects and their labeling information on the current page.

Figure 2-54 Adding a file or deleting a text object

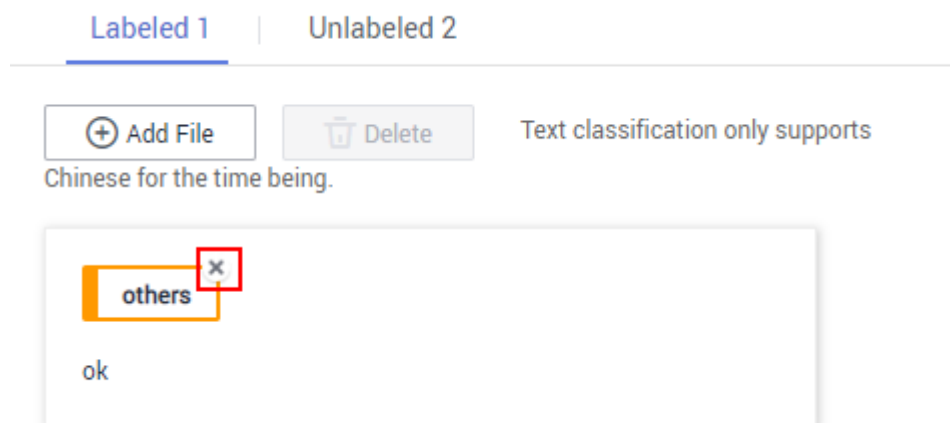


Modifying Labeled Data

For labeled text data, only labels of the text object can be deleted. On the **Labeled** tab page, click the cross icon in the upper right corner of the label to

delete the label of the text object. In the dialog box that is displayed, click **OK**. After the label is deleted, the text object is displayed on the **Unlabeled** tab page.

Figure 2-55 Deleting a labeled text



Modifying a Label

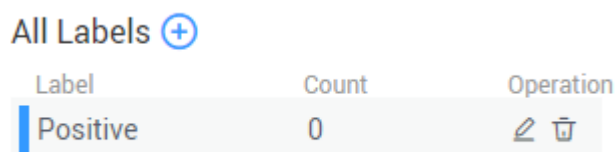
After an ExeML project for text classification is created, you can modify labels based on service changes, including label adding, modification, and deletion.

- Adding a label
On the **Labeled** tab page, click the plus sign (+) on the right of **All Labels**. In the displayed **Add Label** dialog box, set **Label Name** and **Label Color**, and click **OK**.
- Modifying a label
In the lower part of **All Labels** on the **Labeled** tab page, select the label to be modified and click the editing icon in the **Operation** column. In the displayed **Modify Label** dialog box, change the label name or color and click **OK**.
- Deleting a label
In the lower part of **All labels** on the **Labeled** tab page, select a label to be deleted and click the deletion icon in the **Operation** column. In the displayed **Delete** dialog box, select **Delete label** or **Delete the label and objects with only the label**, and click **OK**.

NOTE

The deleted labels cannot be recovered. Exercise caution when performing this operation.

Figure 2-56 Modifying a label



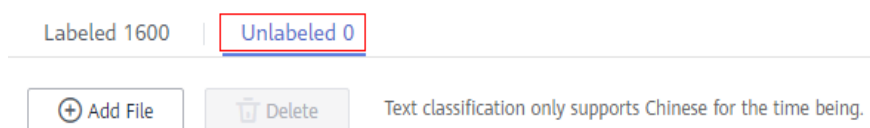
2.6.4 Training a Model

After labeling the data, train a model. You can perform model training to obtain the required text classification model. The text used for training has at least two classifications (that is, more than two labels), and the number of texts in each classification is more than 20. Before training, ensure that the labeled text meets the requirements. Otherwise, the **Train** button is unavailable.

Procedure

1. On the **ExeML** page, click the name of the project that is successfully created. The **Label Data** tab page is displayed.

Figure 2-57 Finding unlabeled files



2. On the **Label Data** tab page, click **Train** in the upper right corner. In the displayed **Training Configuration** dialog box, set related parameters. [Table 2-21](#) describes the parameters.

Figure 2-58 Setting training parameters

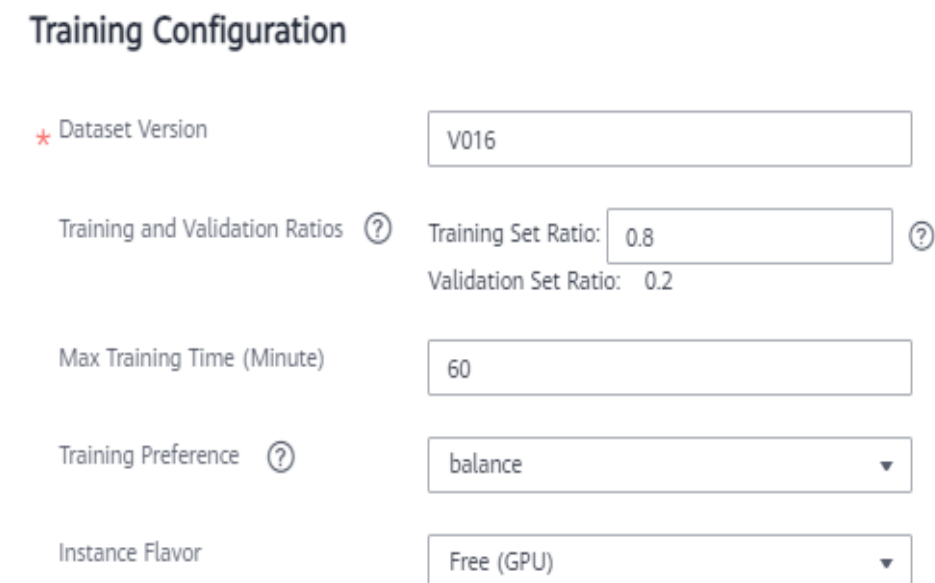


Table 2-21 Parameter description

Parameter	Description	Default Value
Dataset Version	This version is the one when the dataset is published in Data Management . In an ExeML project, when a training job is started, the dataset is published as a version based on the previous data labeling. The system automatically provides a version number. You can change it to the version number that you want.	Randomly provided by the system
Training and Validation Ratios	The labeled sample is randomly divided into a training set and a validation set. By default, the training set ratio is 0.8. The "usage" field in manifest records the set type. The value ranges from 0 to 1.	0.8
Max. Training Duration (Minute)	If the training is not completed within the maximum training duration, the training is forcibly stopped. You are advised to enter a larger value to prevent forcible stop during training. The value ranges from 6 to 6000. Properly extend the training duration. Set the training duration to more than 120 minutes for a training set with 500 pairs of text.	60
Training Preference	<ul style="list-style-type: none"> • performance_first: performance first. The training duration is short and the generated model is small. • balance: balanced performance and precision • accuracy_first: precision first. The training duration is long and the generated model is large. 	balance

Parameter	Description	Default Value
Instance Flavor	<p>Select the resource specifications used for training. By default, the following specifications are supported:</p> <ul style="list-style-type: none"> • Compute-intensive 1 instance (GPU): This flavor is billed on a pay-per-use basis. • Free (GPU): This flavor is free. However, if the flavor is used, the training job automatically stops after one hour. That is, the training job lasts for only one hour at a time. You are advised to evaluate the data size and ensure that the time of a training job does not exceed 1 hour. When there are a large number of users, they need to wait in a queue for this flavor. <p>To use the free flavor, read the message carefully and select I have read and agree to the above.</p>	ExeML (GPU)Compute-intensive 2 instance (NPU)

3. After configuring training parameters, click **Next** to go to the configuration page, confirm the specifications, and click **Submit** to start auto model training. The training takes a certain period of time. Wait until the training is complete. If you close or exit this page, the system still performs the training operation.
4. On the **Train Model** tab page, wait until the training status changes from **Running** to **Completed**.
5. View the training details, such as **Accuracy**, **Evaluation Result**, **Training Parameters**, and **Classification Statistics**. For details about the evaluation result parameters, see [Table 2-22](#).

Figure 2-59 Training details

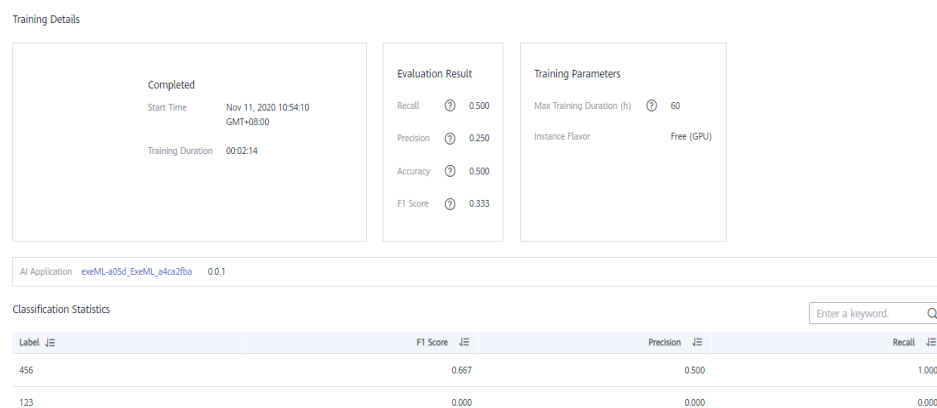


Table 2-22 Evaluation result parameters

Parameter	Description
Recall	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish positive samples.
Precision	Fraction of correctly predicted samples over all samples predicted as a class. It shows the ability of a model to distinguish negative samples.
Accuracy	Fraction of correctly predicted samples over all samples. It shows the general ability of a model to recognize samples.
F1 Score	Harmonic average of the precision and recall of a model. It is used to evaluate the quality of a model. A high F1 score indicates a good model.

 **NOTE**

An ExeML project supports multiple rounds of training, and each round generates a version. For example, the first training version is **V001 (xxx)**, and the next version is **V002 (xxx)**. The trained models can be managed by training version. After the trained model meets your requirements, deploy the model as a service.

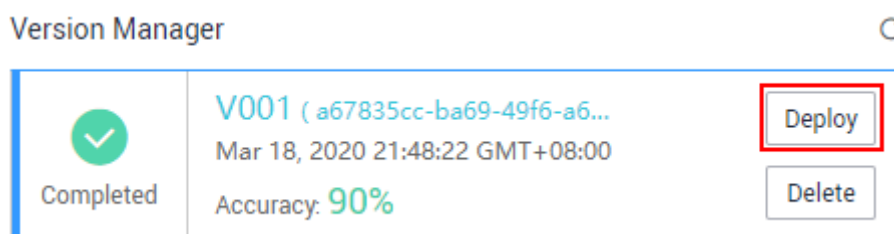
2.6.5 Deploying a Model as a Service

Deploying a Model as a Service

You can deploy a model as a real-time service that provides a real-time test UI and monitoring capabilities. After the model is trained, you can deploy a **Completed** version with ideal accuracy as a service. The procedure is as follows:

1. On the **Train Model** tab page, wait until the training status changes to **Completed**. Click **Deploy** in the **Version Manager** pane to deploy the model as a real-time service.

Figure 2-60 Deploy

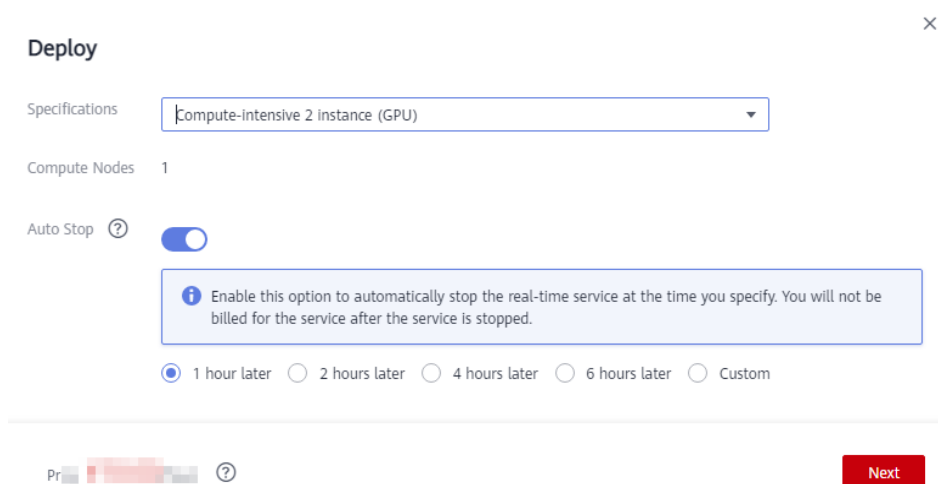


2. In the **Deploy** dialog box, select resource flavor, set the **Auto Stop** function, and click **OK** to start the deployment.
 - **Specifications:** The GPU specifications are better, and the CPU specifications are more cost-effective.

- **Compute Nodes:** The default value is **1** and cannot be changed.
- **Auto Stop:** After this function is enabled and the auto stop time is set, a service automatically stops at the specified time. If this parameter is disabled, a real-time service keeps running and billing. The function can help you avoid unnecessary billing. The auto stop function is enabled by default, and the default value is **1 hour later**.

The options are **1 hour later**, **2 hours later**, **4 hours later**, **6 hours later**, and **Custom**. If you select **Custom**, enter any integer from 1 to 24 in the text box on the right.

Figure 2-61 Deploying a model



3. After the model deployment is started, view the deployment status on the **Service Deployment** page.

It takes a certain period of time to deploy a model. When the status in the **Version Manager** pane changes from **Deploying** to **Running**, the deployment is complete.

NOTE

On the **ExeML** page, trained models can only be deployed as real-time services. For details about how to deploy them as batch services or edge services, see [Where Are Models Generated by ExeML Stored? What Other Operations Are Supported?](#)

Testing a Service

- On the **Service Deployment** page, select a service type. For example, on the ExeML page, the text classification model is deployed as a real-time service by default. On the **Real-Time Services** page, click **Prediction** in the **Operation** column of the target service to perform a service test. For details, see [Testing the Deployed Service](#).
- You can also use code to test a service. For details, see [Accessing Real-Time Services](#).
- The following describes the procedure for performing a service test after the text classification model is deployed as a service on the ExeML page.
 - a. After the model is deployed, you can add a text file for test. On the **ExeML** page, click the target project, go to the **Deploy Service** tab page,

- select the service version in the **Running** status, and enter text to be tested in the text box in the **Service Test** area.
- b. Click **Predict** to perform the test. After the prediction is complete, the result is displayed in the **Test Result** pane on the right. If the model accuracy does not meet your expectation, add text data files on the **Label Data** tab page, label the files, and train and deploy the model again. **Table 2-23** describes the parameters in the prediction result. If you are satisfied with the model prediction result, call the API to access the real-time service as prompted. For details, see [Accessing Real-Time Services](#).

Figure 2-62 Prediction

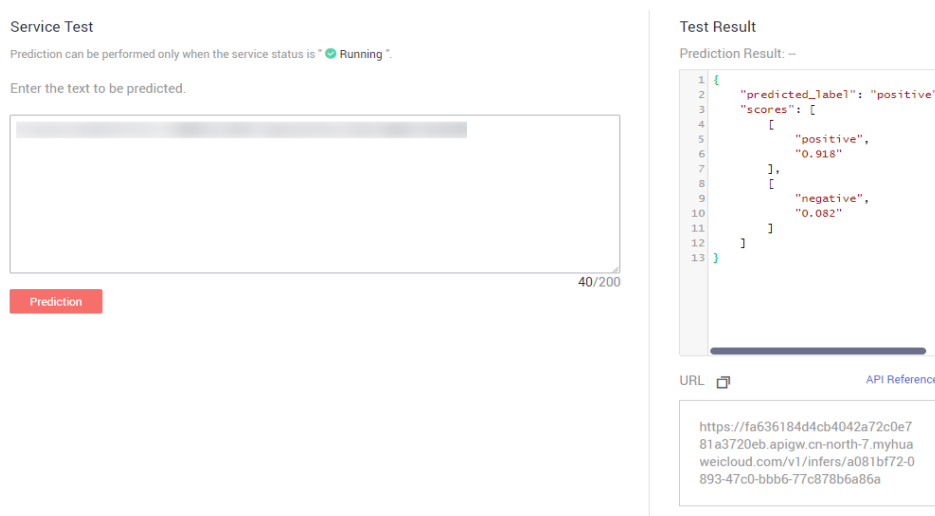


Table 2-23 Parameters in the prediction result

Parameter	Description
predicted_label	Prediction type of the text
score	Confidence score for the predicated class

NOTE

A running real-time service keeps consuming resources. If you do not need to use the real-time service, click **Stop** in the **Version Manager** pane to stop the service so that charges will no longer be incurred. If you want to use the service again, click **Start**.

If you enable the auto stop function, the service automatically stops after the specified time and no fee is generated.

2.7 Tips

2.7.1 How Do I Quickly Create an OBS Bucket and a Folder When Creating a Project?

When creating a project, select a training data path. This section describes how to quickly create an OBS bucket and folder when you select the training data path.


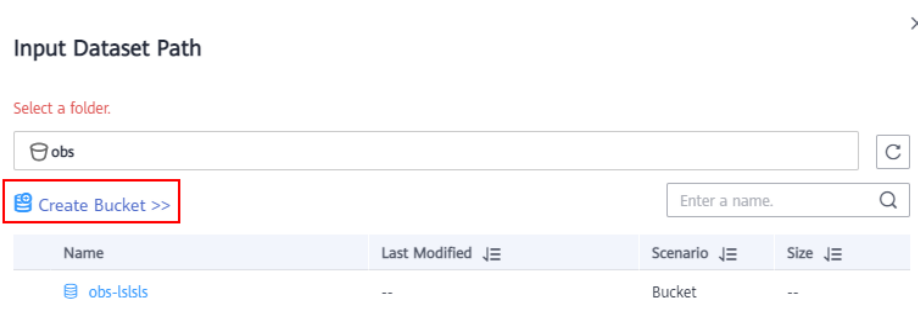
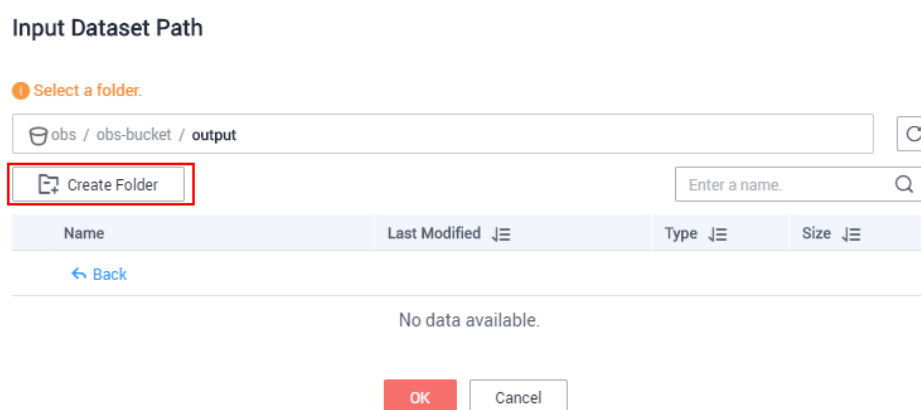
1. On the page for creating an ExeML project, click  on the right of **Input Dataset Path**. The **Input Dataset Path** dialog box is displayed.
2. Click **Create Bucket**. The **Create Bucket** page is displayed. For details, see [Creating a Bucket](#) in *Object Storage Service Console Operation Guide*.

Figure 2-63 Creating an OBS bucket



3. Select the bucket, and click **Create Folder**. In the dialog box that is displayed, enter the folder name and click **OK**.
 - The name cannot contain the following special characters: \/:*?"<>|
 - The name cannot start or end with a period (.) or slash (/).
 - The absolute path of a folder cannot exceed 1,023 characters.
 - Any single slash (/) separates and creates multiple levels of folders at once.

Figure 2-64 Creating a folder



2.7.2 How Do I View the Added Data in an ExeML Project?

To add data for an existing project, perform the following operations. The operations described in this section apply only to sound classification, text classification, object detection and image classification projects. For a predictive analytics project, you can directly add data to its data files.

Obtaining the Data Source of an ExeML Project

1. Log in to the ModelArts management console and choose **ExeML** from the left navigation pane.
2. In the ExeML project list, you can view the data source corresponding to the project in the **Data Source** column. Click your desired data source link to go to the dataset selected or created during project creation.

NOTE

For a predictive analytics project, the data source is an OBS path, not a dataset. For other types of ExeML projects, the data source is a dataset.

Figure 2-65 Viewing the data storage path

Project Name	Project Type	Training Status	Data Source	Created
exeML-voice	Sound Classificati...	Training Failure 2	dataset-voice	Mar 19, 2020 15:52:20 GMT+08:00
exeML-bank	Predictive Analytics	Completed 1	obs://modelarts-test07/Exeml/dat	Mar 19, 2020 15:44:50 GMT+08:00
exeML-yunbao	Object Detection	Completed 1	dataset-yunbao	Mar 18, 2020 22:03:19 GMT+08:00
exeML-flowers	Image Classification	Completed 1	dataset-flowers	Mar 18, 2020 19:39:04 GMT+08:00
exeML-text	Text classification	Training Failure 1	dataset-text	Mar 11, 2020 12:09:44 GMT+08:00

Uploading New Data to OBS

Log in to OBS Console, access the data storage path, and upload new data to OBS.

For details about how to upload files to OBS, see [Uploading an Object](#).

Synchronizing Data to ModelArts

1. After data is uploaded to OBS, go to the **ExeML** page on the ModelArts management console.
2. In the ExeML project list, select the project to which data is to be added and click the project name. The **Label Data** page is displayed.
3. On the **Label Data** page, click **Synchronize Data Source**.

It takes several minutes to complete data synchronization. After the synchronization is complete, the new data is synchronized to the **Unlabeled** or **Labeled** tab page.

2.7.3 How Do I Perform Incremental Training in an ExeML Project?

Each round of training generates a training version in an ExeML project. If a training result is unsatisfactory (for example, if the precision is not good enough), you can add high-quality data or add or delete labels, and perform training again.

NOTE

- Currently, incremental training is only supported for the following types of ExeML projects: image classification, object detection, and sound classification.
- For better training results, use high-quality data for incremental training to improve data labeling performance.

Incremental Training Procedure

1. Log in to the ModelArts console, and click **ExeML** in the left navigation pane.
2. On the **ExeML** page, click a project name. The ExeML details page of the project is displayed.
3. On the **Label Data** page, click the **Unlabeled** tab. On the **Unlabeled** tab page, you can add images or add or delete labels.
If you add images, label the added images again. If you add or delete labels, check all images and label them again. You also need to check whether new labels need to be added for the labeled data.
4. After all images are labeled, click **Train** in the upper right corner. In the **Training Configuration** dialog box that is displayed, set **Incremental Training Version** to the training version that has been completed to perform incremental training based on this version. Set other parameters as prompted.
After the settings are complete, click **Yes** to start incremental training. The system automatically switches to the **Train Model** page. After the training is complete, you can view the training details, such as training precision, evaluation result, and training parameters.

Figure 2-66 Selecting an incremental training version

The screenshot shows the 'Training Configuration' dialog box with the following fields:

- Dataset Version:** V004
- Training and Validation Ratios:** Training Set Ratio: 0.8, Validation Set Ratio: 0.2
- Incremental Training Version:** V002 (highlighted with a red box)
- max training time (minute):** 60
- training preference:** balance
- Instance Flavor:** Compute-intensive 1 instance (GPU)

2.7.4 Where Are Models Generated by ExeML Stored? What Other Operations Are Supported?

Unified Model Management

For an ExeML project, after the model training is complete, the generated model is automatically displayed on the **AI Application Management > AI Applications** page. See the following figure. The model name is automatically generated by the system. Its prefix is the same as the name of the ExeML project for easy identification.

CAUTION

Models generated by ExeML cannot be downloaded.

Figure 2-67 Models generated by ExeML

AI Application Name	Latest Version	Status	Deployment Type
1	0.0.1	Normal	Real-Time Services/Batch Ser...
sv	0.0.2	Normal	Real-Time Services/Batch Ser...
wo	0.0.2	Normal	Real-Time Services/Batch Ser...

What Other Operations Are Supported for Models Generated by ExeML?

- Deploying models as real-time, edge, and batch services**
 On the **ExeML** page, models can only be deployed as real-time services. You can deploy models as batch services or edge services on the **AI Application Management > AI Applications** page.
- Creating a version**
 When creating a new version, you can select a meta model only from a ModelArts training job, OBS, model template, or custom image. You cannot create a version from the original ExeML project.
- Deleting a model or its version**

2.7.5 Upgrading a Project Version

The ExeML function is upgraded to the latest version. If your project is created in the earlier version, you need to upgrade it before using. Data labeling, training, and deployment cannot be performed for ExeML projects that are not upgraded.

NOTE

For Predictive Analytics projects, you do not need to perform the upgrade but directly use the new version of ExeML.

Upgrading ExeML Projects to the Latest Version

- Log in to the ModelArts management console and choose **ExeML** from the left navigation pane. The ExeML list page is displayed.
- Find the projects of the earlier version. In the ExeML project list, if the project is of an earlier version, the project name is marked with a tag. For such a project, click **upgrade** in the **Operation** column.

If your projects are of the latest version, the **upgrade** button is not displayed in the **Operation** column.

Figure 2-68 Project of the earlier version

Note: Upgrade your ExeML projects to the latest version before using them.

Project Name	Project Type	Training Status	Data Source	Created	Description	Operation
exeML-84d6	Image Classificat...	Model Publish Failed 2	dataset-Special	Dec 18, 2019 16:33:56 GMT+08:00	--	Delete
exeML-c737	Text classification	Completed 4 Model Pub...	dataset-eaf2	Dec 18, 2019 16:04:31 GMT+08:00	--	Delete
exeML-70f1	Sound Classificat...	Completed 4	obs://getobs/voice/	Dec 05, 2019 09:43:53 GMT+08:00	this is a project migrated fro...	Update Delete


3. In the dialog box that is displayed, set **Dataset Name** and **Storage path** of the dataset to be saved, and click **Yes** to start the upgrade.

Figure 2-69 Upgrading a project

Upgrade ProjectexeML-70f1

* Dataset Name

* Storage path

 Select an OBS path that is not under the /getobs/voice/ directory.

Wait for the project upgrade. After several minutes, the project is upgraded to the latest version. The earlier version tag is not displayed in the **Project Name** column, and the **upgrade** button is not displayed in the **Operation** column.