# 异常检测（空气处理单元） - 模型推断

本Jupyter笔记本演示了如何使用SAS事件流处理（ESPPy）模块执行使用之前创建并存储在分析存储文件中的模型进行异常检测。

关于此用例的更多资源可以在[SAS GitHub页面](https://github.com/sassoftware/iot-anomaly-detection-hvac)找到，其中包括在SAS事件流处理工作室中部署离线模型的示例。

### 0. 设置环境

首先，导入运行此笔记本所需的包。设置主目录，告诉笔记本在哪里保存模型XML文件。

```python

import esppy

import ipywidgets as widgets

from esppy.espapi.visuals import Visuals

from inspect import getsource

import time

import datetime

%matplotlib inline

import pandas as pd

```

使用Pandas读取数据。

```python

ahu\_data = pd.read\_csv('/demo/Event\_Stream\_Processing/data/ahu\_scr.csv', header=0)

```

将<code>display.image\_scale</code>的值设置为0.75。这个值可以帮助你更好地可视化项目，随着添加更多窗口。

```python

esppy.options.display.image\_scale = 0.75

```

接下来，为项目建立与ESP服务器的连接，还有一个用于PLOTLY图形库API的额外连接。

```python

esp = esppy.ESP('http://localhost:5001')

conn = esp.createServerConnection(interval=0)

esp

```

### 1. 创建项目

使用<code>esp.create\_project</code>创建一个名为\*\*proj\*\*的项目。

```python

proj = esp.create\_project('AHU\_AnomalyDetection')

```

现在创建一个连续查询并命名为<code>cq</code>，将此查询添加到项目中。连续查询会自动定期在流数据上运行。

```python

cq = esp.ContinuousQuery(name='Query\_1')

proj.add\_query(cq)

```

### 2. 创建项目窗口

首先，创建一个源窗口并命名为<code>src\_data</code>。源窗口接受流数据或原始数据文件。

```python

src = esp.SourceWindow(schema=('Key\_ID\*:int64', 'seq\_id:string', 'datetime:string',

'AHU:string', 'CHW\_VALVE:double', 'CHW\_VALVE\_POSIT:double',

'DIS\_AIR\_TEMP:double', 'DUCT\_PRESS\_ACTV:double', 'MAX\_CO2\_VAL:double',

'MIXED\_AIR\_TEMP:double', 'RTRN\_AIR\_TEMP:double', 'SUPPL\_FAN\_SP:double'),

index\_type='empty', insert\_only=True, name='src\_data')

cq.add\_window(src)

```

创建一个用于读取存储在模型数据中的模型数据的源窗口。

```python

model\_request = esp.SourceWindow(schema=('req\_id\*:int64', 'req\_key:string', 'req\_val:string'),

index\_type='empty', insert\_only=True, name='model\_request')

cq.add\_window(model\_request)

```

接下来，创建一个窗口来读取存储在分析存储文件中的模型。命名为<code>model\_reader</code>。

```python

model\_reader = esp.ModelReaderWindow(name='Model\_Reader', model\_type="astore")

cq.add\_window(model\_reader)

```

创建一个Score窗口，使用离线分析存储模型对\*\*ahu\_scr\*\*数据进行评分。

```python

score\_svdd = esp.ScoreWindow(name='Score\_SVDD',

schema=('Key\_ID\*:int64', 'seq\_id:string', 'datetime:string', 'AHU:string',

'\_SVDDDISTANCE\_:double', '\_SVDDSCORE\_:double'))

score\_svdd.add\_connector('fs', conn\_name='sub', conn\_type='subscribe',

properties={'type':'sub', 'fstype':'csv', 'fsname': '/user/my\_data/svdd\_out.csv', 'snapshot':True,'header':'full'})

score\_svdd.add\_offline\_model(model\_type='astore')

cq.add\_window(score\_svdd)

```

窗口创建并添加到项目后，可以通过创建边缘来连接它们。关于连接器的更多信息，请参阅[连接器概述](https://go.documentation.sas.com/?cdcId=espcdc&cdcVersion=6.2&docsetId=espca&docsetTarget=p1nhdjrc9n0nnmn1fxqnyc0nihzz.htm&locale=en)。

```python

src.add\_target(score\_svdd, role='data')

model\_request.add\_target(model\_reader, role='request')

model\_reader.add\_target(score\_svdd, role='model')

```

### 3. 将项目保存为XML并加载到ESP服务器

可以选择将创建的项目保存为XML文件。在这里，将'AHUAnomalyDetection.xml'保存到/shared/code文件夹中。

```python

proj.save\_xml('/user/my\_code/ahu\_anomaly\_detection.xml')

```

然后使用<code>esp.load\_project</code>将项目加载到ESP服务器，并将项目代码保存到/shared/code文件夹中。

```python

esp.load\_project(proj)

```

### 4. 订阅项目窗口

订阅项目窗口以接收它们的输出。

```python

src.subscribe()

model\_request.subscribe()

score\_svdd.subscribe()

model\_reader.subscribe()

```

```python

ahuScoring = conn.getEventCollection("AHU\_AnomalyDetection/Query\_1/Score\_SVDD")

```

### 5. 发布离线模型并加载数据进行评分

创建一个发布器，将模型发布到score\_svdd窗口，以便项目知道要使用哪个模型。

这一步大约需要20秒钟。包含<code>time.sleep(20)</code>指令以确保在发送图像到服务器之前建立连接。

```python

pubmodel = model\_request.create\_publisher(blocksize=1, rate=0, pause=0,

opcode='insert', format='csv')

pubmodel.send('i,n,1,"action","load"\n')

pubmodel.send('i,n,2,"type","astore"\n')

pubmodel.send('i,n,3,"reference","/demo/Event\_Stream\_Processing/data/ahu\_svdd.astore"\n')

pubmodel.send('i,n,4,,\n')

pubmodel.close()

time.sleep(5)

```

将原始数据变量分配给一个变量。

```python

mydata = ahu\_data

mydata.index.name = 'Key\_ID'

mydata

```

使用<code>publish\_events()</code>方法将数据发布到模型的数据事件源窗口。<code>subscribe()</code>方法创建一个DataFrame作为Source窗口的<code>data</code>属性。在窗口对象的<code>data</code>属性中创建DataFrame后，可以像操作DataFrame一样使用Pandas DataFrame方法。

```python

src.publish\_events(mydata, pause=1000)

src.subscribe()

src.info()

model\_reader

```

### 6. 显示评分结果

通过订阅'score\_svdd'窗口输出来显示评分结果。

```python

score\_svdd

```

可选地，使用PLOTLY图形库的接口以自定义格式显示结果。

```python

conn = esp.createServerConnection()

sf\_input = conn.getEventStream("AHU\_AnomalyDetection/Query\_1/Score\_SVDD", maxevents=20)

visuals = Visuals(colormap="sas\_corporate", border="1px solid blue")

table = visuals.createTable(sf\_input, values=["Key\_ID", "\_SVDDDISTANCE\_", "\_SVDDSCORE\_"], title="Values Generated",

show\_controls=True, width="35%")

chart = visuals.createBubbleChart(sf\_input, y="\_SVDDDISTANCE\_", size="\_SVDDDISTANCE\_", color="\_SVDDSCORE\_",

title="SVDD Distance")

widgets.HBox([table, chart])

```

这些步骤演示了如何使用SAS Event Stream Processing ESPPy模块进行实时异常检测，使用事先创建并存储在分析存储文件中的模型。