Anomaly Detection Service : Hands On I

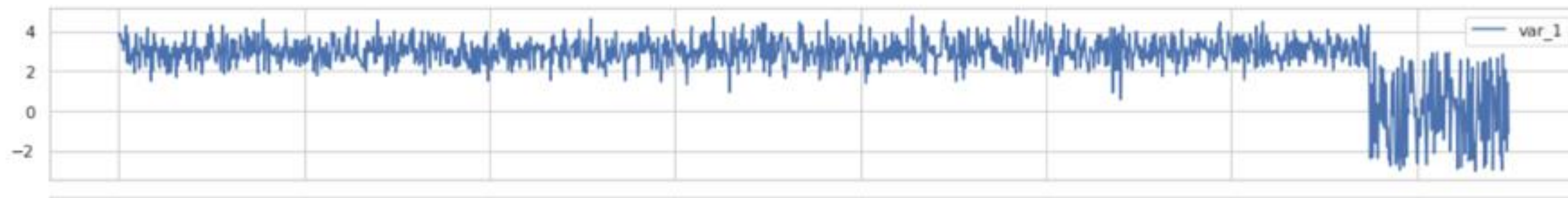
Presenter : Dr. Dhaval Patel
pateldha@us.ibm.com

Outline

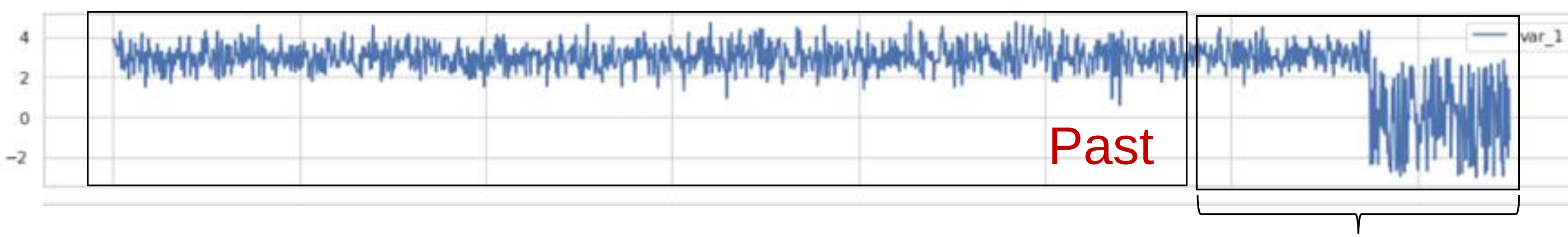
- ❑Unsupervised AD in Univariate Time Series
- ❑Unsupervised AD in Multivariate Time Series
- ❑Regression-Model based AD
- ❑Semi-supervised AD
- ❑Mixture-Model based AD

Unsupervised Anomaly Detection Scenario

- ***Batch/Entire*** : Detect Anomaly in Given time series

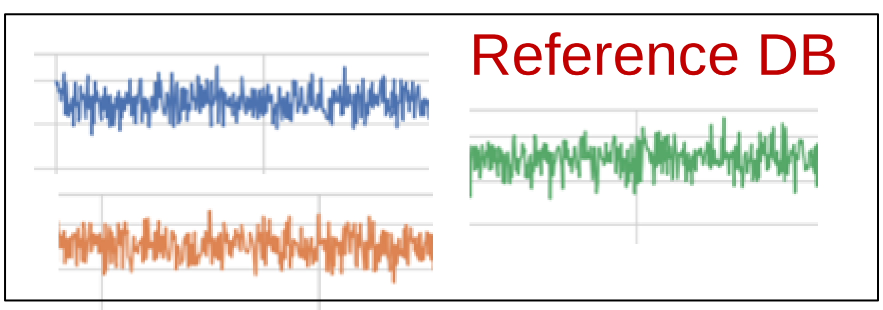


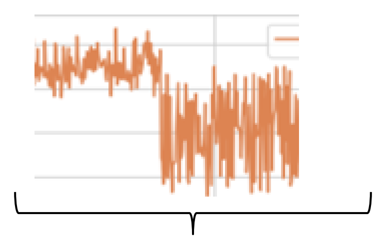
- ***Stream/Recent*** : Use Past time series to detect anomaly in recent input



Detect Anomaly Here

- ***Across*** : Use other time series to detect anomaly in given time series





Detect Anomaly Here

Unsupervised Anomaly Detection API

API end points

https://developer.ibm.com/apis/catalog/ai4industry--anomaly-detection-product/api/API--ai4industry--anomaly-detection-api#batch_uni

Overview
- Introduction
- Getting Started

</> Anomaly Detection API
- Connection Check
- Get Result
- Submit an Anomaly Detection Job
  - Detect anomalies on multivariate time series
  - **Detect anomalies on univariate time series**
  - Discover anomaly model using semi-supervised approach
  - Discover mixture model based anomaly
  - Discover regression based anomaly model

</> Code snippet

Detect anomalies on univariate time series

cURL | Ruby | Python | PHP | Java | Node | Go | Swift

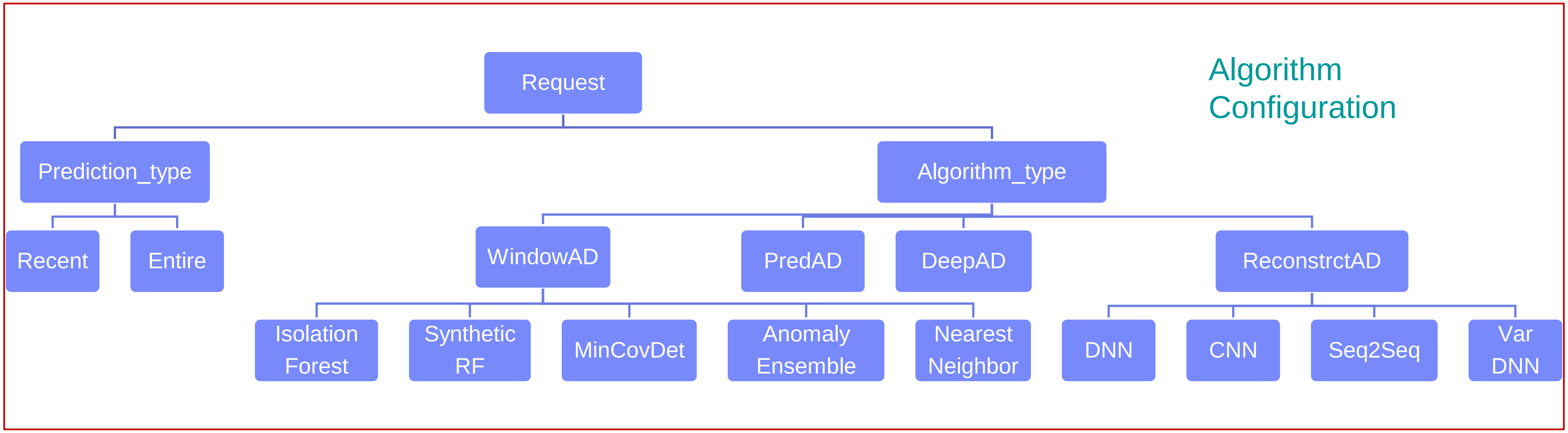
```
curl --request POST \
  --url https://api.ibm.com/ai4industry/run/anomaly-detection/timeseries/univariate/batch \
  --header 'X-IBM-Client-Id: REPLACE_THIS_KEY' \
  --header 'X-IBM-Client-Secret: REPLACE_THIS_KEY' \
  --header 'accept: application/json' \
  --header 'content-type: multipart/form-data' \
  --form data_file=REPLACE_THIS_VALUE \
  --form time_column=REPLACE_THIS_VALUE \
  --form time_format=REPLACE_THIS_VALUE \
  --form target_column=REPLACE_THIS_VALUE \
  --form prediction_type=REPLACE_THIS_VALUE \
  --form recent_data=REPLACE_THIS_VALUE \
  --form algorithm_type=REPLACE_THIS_VALUE \
  --form anomaly_estimator=REPLACE_THIS_VALUE \
  --form lookback_window=REPLACE_THIS_VALUE \
  --form observation_window=REPLACE_THIS_VALUE \
  --form labeling_method=REPLACE_THIS_VALUE \
  --form labeling_threshold=REPLACE_THIS_VALUE
```

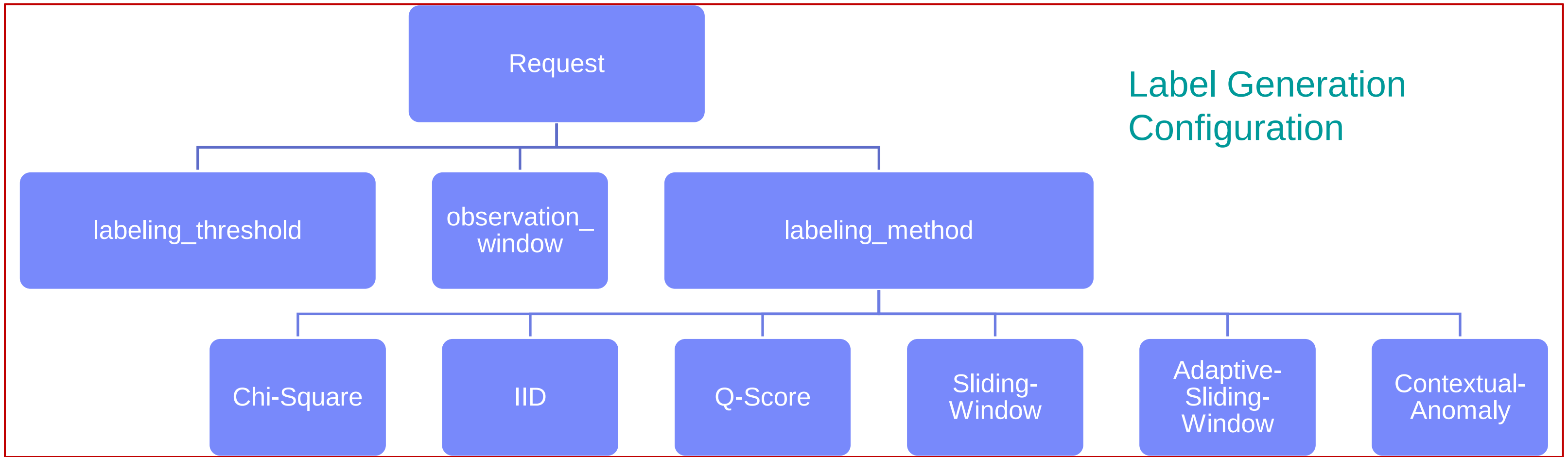
OpenAPI doc

Try this API

Unsupervised Anomaly Detection API

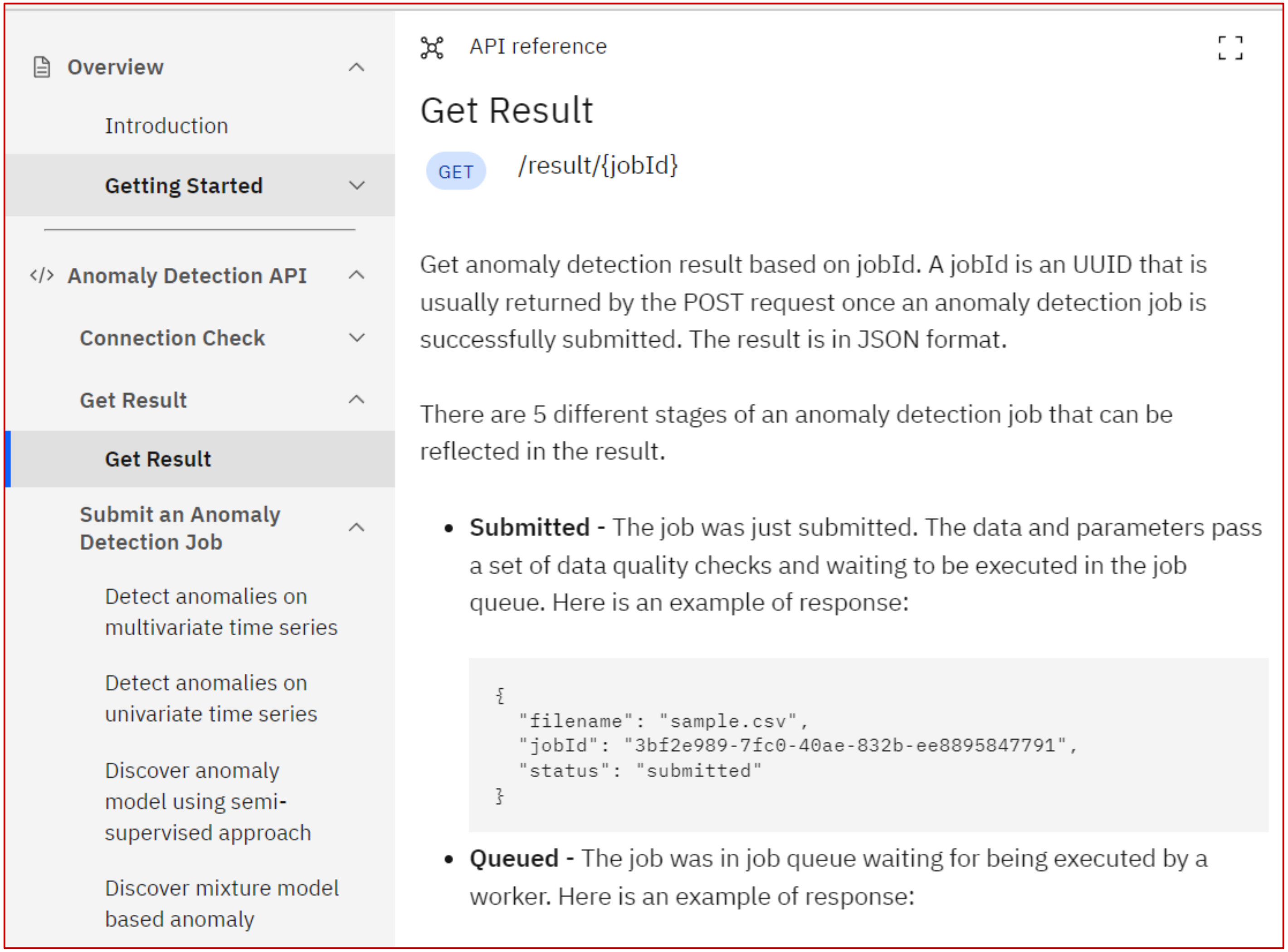
- API arguments are categorized into four configuration
  – Meta Data : target column, feature column, time column and format, etc
  – Algorithm Configuration : Which algorithm to run
  – Evaluation Setting : Instance size, evaluation metric, etc
  – Anomaly Label Generation : How to generate anomaly label (+1/-1)





Unsupervised Anomaly Detection Execution

- On submitting a user request, AD service return a “Job ID”
- End user can use : https://developer.ibm.com/apis/catalog/ai4industry--anomaly-detection-product/api/API--ai4industry--anomaly-detection-api#get_result_by_id



Overview

Introduction

Getting Started

</> Anomaly Detection API

Connection Check

Get Result

Get Result

Submit an Anomaly Detection Job

Detect anomalies on multivariate time series

Detect anomalies on univariate time series

Discover anomaly model using semi-supervised approach

Discover mixture model based anomaly

API reference

Get Result

GET /result/{jobId}

Get anomaly detection result based on jobId. A jobId is an UUID that is usually returned by the POST request once an anomaly detection job is successfully submitted. The result is in JSON format.

There are 5 different stages of an anomaly detection job that can be reflected in the result.

- **Submitted -** The job was just submitted. The data and parameters pass a set of data quality checks and waiting to be executed in the job queue. Here is an example of response:

```
{
  "filename": "sample.csv",
  "jobId": "3bf2e989-7fc0-40ae-832b-ee8895847791",
  "status": "submitted"
}
```

- **Queued -** The job was in job queue waiting for being executed by a worker. Here is an example of response:

Unsupervised Anomaly Detection Execution

- Each Job in processed by Celery Work
- Celery worker submit job to Code Engine (Serverless Computing Capability)
- Code Engine run an Anomaly Workflow for each request

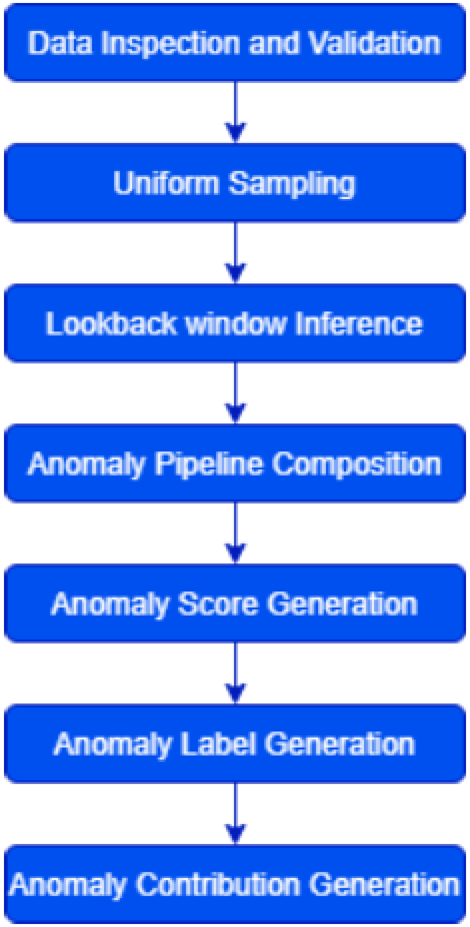


Figure 2: Anomaly Workflow

Unsupervised Anomaly Detection Execution

- Anomaly Score and Anomaly Label are two important outcome

```
result = []
result_header = ['timestamp', 'anomaly_score', 'anomaly_label']

for item in json_data['summary']['result']:
    result.append([item['timestamp'], item['value']['anomaly_scor

result = pd.DataFrame(result)
result.columns = result_header
result.tail(10)
```

	timestamp	anomaly_score	anomaly_label
990	2017-01-04 10:30:00	0.000154	1.0
991	2017-01-04 10:35:00	0.009484	1.0
992	2017-01-04 10:40:00	0.013656	1.0
993	2017-01-04 10:45:00	0.020583	1.0
994	2017-01-04 10:50:00	0.540153	1.0
995	2017-01-04 10:55:00	0.594477	1.0
996	2017-01-04 11:00:00	0.438216	1.0
997	2017-01-04 11:05:00	1.541768	1.0
998	2017-01-04 11:10:00	-0.000458	1.0
999	2017-01-04 11:15:00	0.372137	1.0

Demo 1

- Clone repo :

https://github.com/IBM/anomaly-detection-code-pattern

- Run Notebook (Expected execution time ~ 1 minutes) –

https://github.com/IBM/anomaly-detection-code-pattern/blob/main/notebooks/Univariate_AD_service_sample_data.ipynb

```
file_path = './datasets/univariate/sample_data/' + datafile_name
files = {'data_file': (datafile_name, open(file_path, 'rb'))}

data = {
    'target_column': value,
    'time_column': timestamp,
    'time_format': time_format,
    'prediction_type': 'entire',
    'algorithm_type': 'DeepAD',
    'lookback_window': 'auto',
    'observation_window': 10,
    'labeling_method': 'Chi-Square',
    'labeling_threshold': 10,
    'anomaly_estimator': 'Default',
}

headers = {
    'X-IBM-Client-Id': Client_ID,
    'X-IBM-Client-Secret': Client_Secret,
    'accept': "application/json",
    }

import requests
post_response = requests.post("https://api.ibm.com/ai4industry/run/anomaly-detection/timeseries/univariate/batch",
                              data=data,
                              files=files,
                              headers=headers)
```

```
--------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------
lookback_window             33
model_summary               [('SkipTransformer',NoOp()),('NormalizedFlatten',NormalizedFlatten(feature_columns=[1],lookback_win=33,t
arget_columns=[1])),('LinearSVR',LinearSVR(random_state=0)),]
num_pipelines_explored      75
total_execution_time (sec)  54.34661364555359
```

Demo 2

- Clone repo :

https://github.com/IBM/anomaly-detection-code-pattern

- Run Notebook with Modification (Expected execution time ~ 1 minutes) –

https://github.com/IBM/anomaly-detection-code-pattern/blob/main/notebooks/Univariate_AD_service_sample_data.ipynb

Algorithm Type	Anomaly Estimator	Execution Time	Observations
WindowAD	MinCovDet	46 Sec	??
WindowAD	IsolationForest	21 Sec	
WindowAD	AnomalyEnsembler	23 Sec	
ReconstructAD	CNN_AutoEncoder	28 Sec	
ReconstructAD	Seq2seq_AutoEncoder	40 Sec	

!!! Look at the Anomaly Score
!!! Look at the Model Summary
!!! Vary Labelling Method

Demo 3

- Clone repo :

  https://github.com/IBM/anomaly-detection-code-pattern

- Run Other Notebook

  https://github.com/IBM/anomaly-detection-code-pattern/blob/main/notebooks/Univariate_AD_service_public_data.ipynb

Dataset	Num of Records	Algorithm Used
ec2_network_in_5abac7	4718	ReconstructAD
Twitter_volume_AAPL	150902	DeepAD
Bitcoin price	30066	PredAD

Testing Support for Long Time Series

Unsupervised Anomaly Detection in Multivariate Time Series

IBM

- Build anomaly model using multiple time series as input
- API end points
  https://developer.ibm.com/apis/catalog/ai4industry--anomaly-detection-product/api/API--ai4industry--anomaly-detection-api#batch_uni

Overview

Introduction

Getting Started

</> Anomaly Detection API

Connection Check

Get Result

Submit an Anomaly Detection Job

Detect anomalies on multivariate time series

Detect anomalies on univariate time series

Discover anomaly model using semi-supervised approach

</> Code snippet

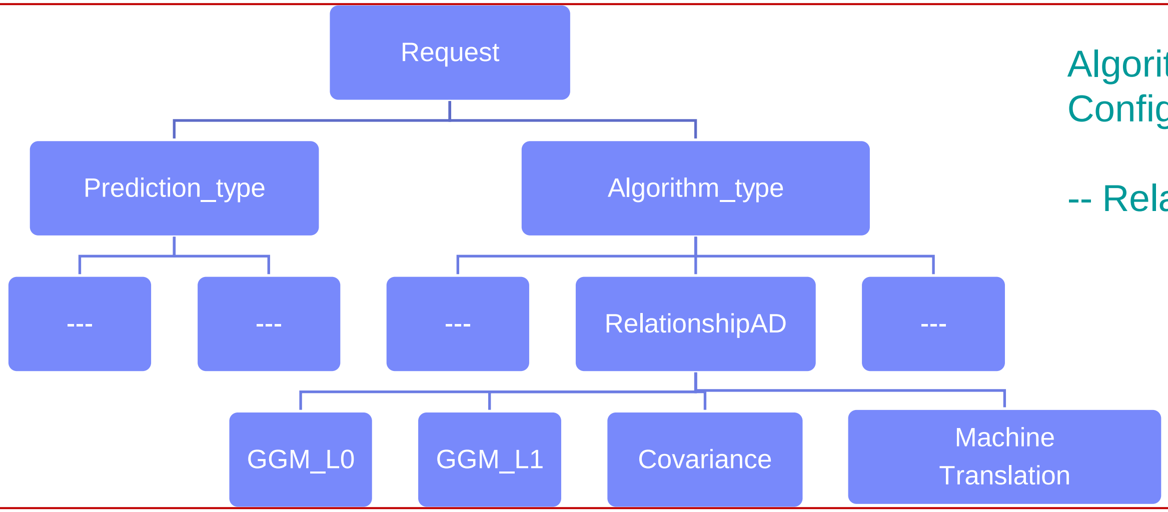
Detect anomalies on multivariate time series

cURL | Ruby | Python | PHP | Java | Node | Go | Swift

```
curl --request POST \
  --url https://api.ibm.com/ai4industry/run/anomaly-detection/timeseries/multivariate/batch \
  --header 'X-IBM-Client-Id: REPLACE_THIS_KEY' \
  --header 'X-IBM-Client-Secret: REPLACE_THIS_KEY' \
  --header 'accept: application/json' \
  --header 'content-type: multipart/form-data' \
  --form data_file=REPLACE_THIS_VALUE \
  --form time_column=REPLACE_THIS_VALUE \
  --form time_format=REPLACE_THIS_VALUE \
  --form target_columns=REPLACE_THIS_VALUE \
  --form prediction_type=REPLACE_THIS_VALUE \
  --form recent_data=REPLACE_THIS_VALUE \
  --form algorithm_type=REPLACE_THIS_VALUE \
  --form anomaly_estimator=REPLACE_THIS_VALUE \
  --form lookback_window=REPLACE_THIS_VALUE \
  --form observation_window=REPLACE_THIS_VALUE \
  --form labeling_method=REPLACE_THIS_VALUE \
  --form labeling_threshold=REPLACE_THIS_VALUE
```

- API arguments are categorized into four configuration
  – Meta Data : target column, feature column, time column and format, etc
  – Algorithm Configuration : Which algorithm to run
  – Evaluation Setting : Instance size, evaluation metric, etc
  – Anomaly Label Generation : How to generate anomaly label (+1/-1)

- Algorithm Configuration : All that we discussed for Univariates and then ***RelationshipAD***



Algorithm Configuration

-- RelationshipAD

Demo 4

- Clone repo :

  https://github.com/IBM/anomaly-detection-code-pattern

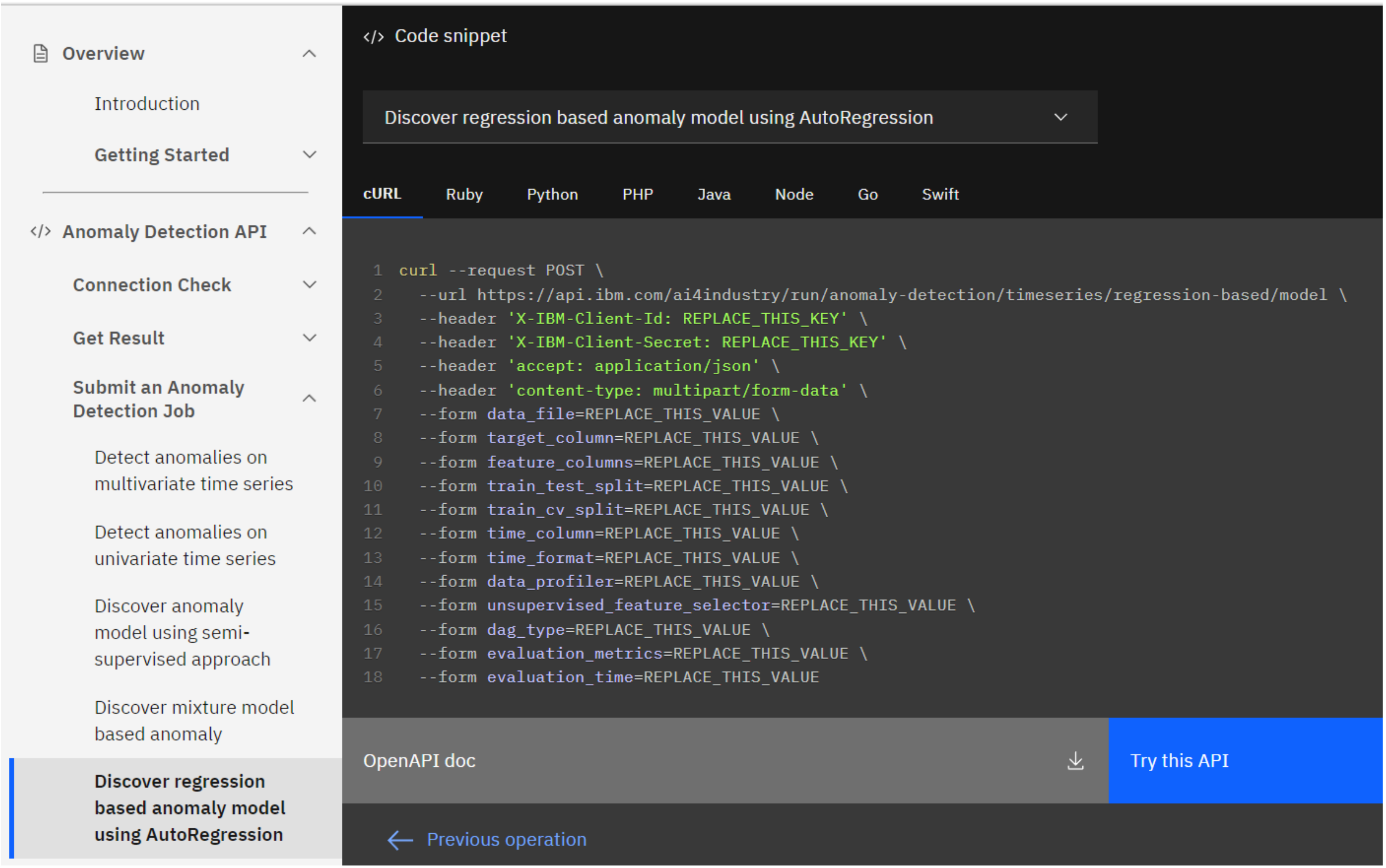
- Run Other Notebook

  https://github.com/IBM/anomaly-detection-code-pattern/blob/main/notebooks/Multivariate_AD_service_sample_data.ipynb

Regression-aware Anomaly Detection API

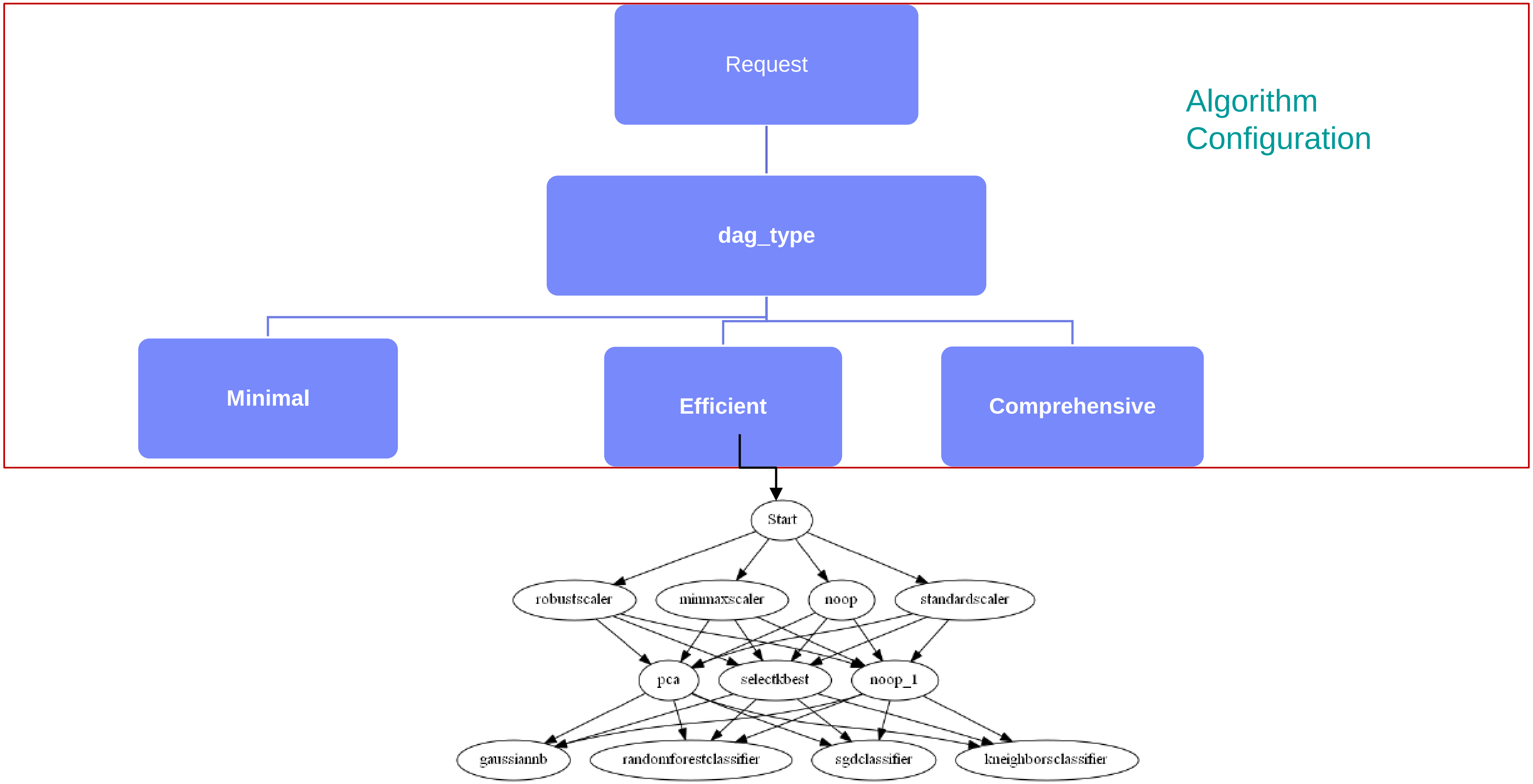
API end points

https://developer.ibm.com/apis/catalog/ai4industry--anomaly-detection-product/api/API--ai4industry--anomaly-detection-api#model_regression_based



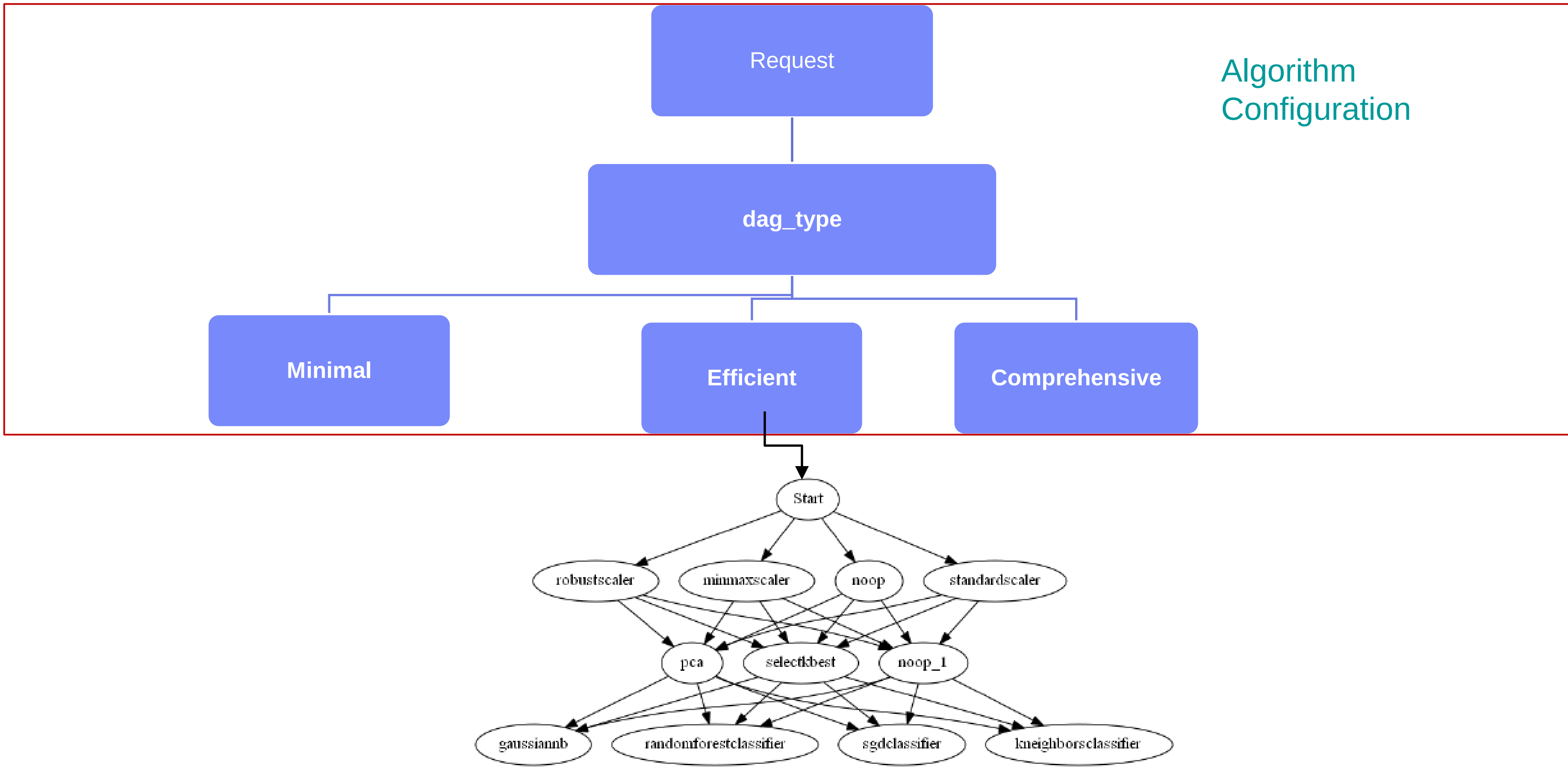
Regression-aware Anomaly Detection API

- Important API arguments
  - target column
  - feature column
  - Directed Acyclic Graph based ***AutoRegression***



Regression-aware Anomaly Detection API

- Important API arguments
  - Model Selection Process
  - **train_test_split -** partition a data into train and test. train dataset is used for model ranking. Test dataset is used for hold out evaluation and prediction output
  - **train_cv_split** – model ranking



Demo 5

- Clone repo :

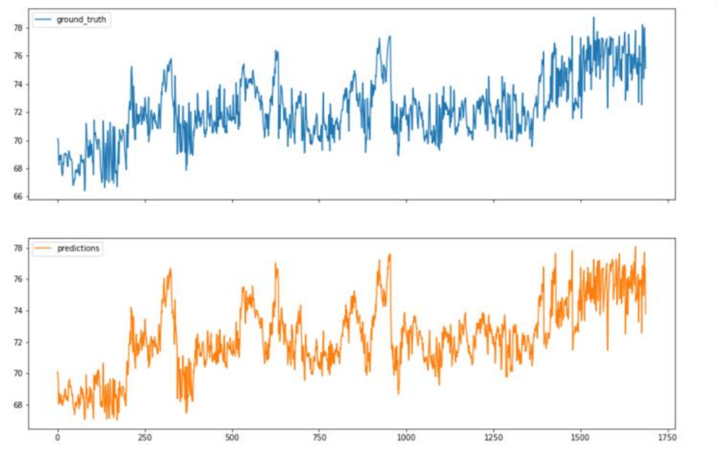
  https://github.com/IBM/anomaly-detection-code-pattern

- Run Other Notebook

  https://github.com/IBM/anomaly-detection-code-pattern/blob/main/notebooks/Regression-aware_AD_service_sample_data.ipynb

In [36]:
```
import numpy as np

time_column = "Date_time"
feature_columns = ['Gost_avg', 'Git_avg', 'Rs_avg', 'Ws_avg', 'Wa_avg']
target_columns = 'Gb1t_avg'
time_format="%Y-%m-%d %H:%M:%S"
```



Unsupervised Feature Selection

- A Boolean argument in many Anomaly Detection APIs
- The feature selection is motivated from many IoT applications that monitors the same process/asset using redundant set of the sensor variable
  - Data driven discovery of the process
- Apply ensemble of three feature selectors
  - Variance based
  - Influence Factor
  - Correlation based
- If more than two feature Selector tag an input columns
  - We drop the column from further analysis
- Speed-up related optimization
  - We have embedded the optimization related to the size of data and number of records on increasing the speed on the feature related task

Obtain IBM Cloud Access

- https://www.ibm.com/cloud/object-storage/faq