NEC Advanced Analytics - RAPID 機械学習 時系列数値解析 V2.2 ユーザガイド

第 1.1 版
2018 年 12 月

日本電気株式会社

はしがき

本書は、「NEC Advanced Analytics - RAPID 機械学習 時系列数値解析 for Windows V2.2」、「NEC Advanced Analytics - RAPID 機械学習 時系列数値解析 for Linux V2.2」（以降、本製品と記載）の利用者のための説明書です。

本製品の利用方法について解説しています。本製品の利用前に、必ずお読みください。

また、本製品に関する説明書は、セットアップ媒体の Document フォルダに格納されています。必要に応じて、参照してください。

＜商標について＞

- Linux は、米国およびその他の国における Linus Torvalds の登録商標です。
- Red Hat は、米国 Red Hat, Inc.ならびにその子会社の登録商標です。
- Windows、Hyper-V は、米国 Microsoft Corporation の米国およびその他の国における商標または登録商標です。
- Intel、Intel Core、Xeon は、米国 Intel 社の商標または登録商標です。
- NVIDIA、Tesla は、米国 NVIDIA Corporation の米国およびその他の国における商標または登録商標です。
- その他、本書に記載されている会社名、製品名は、一般に各社の商標または登録商標です。

＜略語・用語について＞

- Windows の正式名称は、Microsoft Windows Operating System です。

＜輸出する際の注意事項＞

本製品（ソフトウェアを含む）は、外国為替および外国貿易法で規定される規制貨物（または役務）に該当することがあります。 その場合、日本国外へ輸出する場合には日本国政府の輸出許可が必要です。 なお、輸出許可申請手続にあたり資料等が必要な場合には、お買い上げの販売店またはお近くの当社営業拠点にご相談ください。

更新履歴

版数	日付	改版内容
1.0 版	2018 年 10 月 1 日	初版
1.1 版	2018 年 12 月 18 日	学習コンフィグファイルの 1DOCN のパラメータ設定方針に関する誤記を修正。

目次

第 1 章 はじめに 6

1.1 用語定義 6

第 2 章 本製品について 8

2.1 システム構成 8

2.2 モジュール概要 8

第 3 章 使用方法 9

3.1 時系列数値データ分析の流れ 9

3.1.1 学習データ作成フェーズ 9

3.1.2 学習フェーズ 9

3.1.3 予測フェーズ 9

3.2 学習と予測について 10

3.2.1 分類 10

3.2.2 回帰 10

3.2.3 OneClass 分類 11

第 4 章 CLI 使用例 13

4.1 分類の使用例 13

4.1.1 学習データ生成フェーズ 13

4.1.2 学習フェーズ 14

4.1.3 予測フェーズ 16

4.2 回帰の使用例 19

4.2.1 学習データ生成フェーズ 20

4.2.2 学習フェーズ 21

4.2.3 予測フェーズ 23

4.3 OneClass 分類の使用例 26

4.3.1 学習データ生成フェーズ 26

4.3.2 学習フェーズ 27

4.3.3 予測フェーズ 29

4.4 予測のみ実施する場合 34

4.4.1 予測モデルの移行 34

4.4.2 予測フェーズ 35

第 5 章 CLI 詳細 37

5.1 CLI 一覧 37

5.1.1 統計情報取得コマンド（exec_calc_stats） 37

5.1.2 前処理コマンド（exec_preprocess） 39

5.1.3 学習コマンド（exec_train） 42

5.1.4 予測コマンド（exec_predict） ····· 45
5.1.5 オンライン予測 ····· 47
5.2 使用可能文字種別 ····· 56
5.3 Exit コード一覧 ····· 57

第 6 章 ファイル ····· 58
6.1 時系列数値データファイル ····· 58
6.2 ラベルファイル ····· 59
6.3 結果ファイル ····· 61
6.3.1 簡易予測結果ファイル ····· 61
6.3.2 詳細予測結果ファイル ····· 62
6.4 前処理定義ファイル ····· 63
6.4.1 フォーマット ····· 63
6.4.2 処理一覧 ····· 64
6.5 統計情報ファイル ····· 66
6.6 コンフィグファイル ····· 66
6.6.1 実行コンフィグファイル ····· 66
6.6.2 学習コンフィグファイル ····· 68
6.6.3 予測コンフィグファイル ····· 72
6.6.4 オンライン予測コンフィグファイル ····· 75
6.6.5 CLI ログコンフィグファイル ····· 77
6.6.6 MLE ログコンフィグファイル ····· 79
6.7 ログファイル ····· 81

第 7 章 注意・制限事項 ····· 83
7.1 注意事項 ····· 83
7.1.1 本製品の使用時に使用するユーザ権限 ····· 83
7.1.2 実行コンフィグファイルの更新に使用するユーザ権限 ····· 83
7.1.3 前処理コマンド実行後の学習・予測コマンドの実行について ····· 83
7.1.4 1DOCN の学習における処理時間について ····· 83
7.1.5 学習実行中のコンフィグファイルの編集について ····· 83
7.2 制限事項 ····· 83

第1章 はじめに

本書は、本製品の利用に関する説明書です。

本章では、本書を読む上で必要となる前提知識を説明します。

1.1 用語定義

本書で使用する用語を『表 1-1』で定義します。

表 1-1 本書で使用する用語の説明

用語・略語	説明
RAPID 機械学習	NEC 独自のディープラーニング技術を実装した機械学習手法、および機械学習アプリケーションです。
RAPID 機械学習 エンジン	RAPID 機械学習の実行基盤です。
ニューラルネットワーク	機械学習手法の一種で、人間の脳神経細胞の仕組みを模した手法です。ニューラルネットとも呼ばれます。主に予測データの分類や回帰、クラスタリングに用いられます。
ディープラーニング	構造が複雑かつ大規模なニューラルネットワークです。データから特徴量を計算する方法を学習できるため、通常のニューラルネットワークより高精度な解析が実現できます。
RAPID 機械学習 時系列数値解析	RAPID 機械学習 エンジンを使用して、時系列数値解析を行うための製品です。
TSA	Time Series data Analysis の略で、RAPID 機械学習 時系列数値解析の略称です。
CLI	本書では、RAPID 機械学習 時系列数値解析の製品に含まれるコマンドラインインタフェースのことを指します。 統計情報取得機能、前処理機能、学習機能、予測機能、オンライン予測機能を提供しています。
Time Series Marker	RAPID 機械学習 時系列数値解析の製品に含まれる、時系列数値データに対するラベルファイル作成を支援する GUI です。
時系列数値データ	ある現象の時間的な変化を観測して得られた数値の列です。
1D CNN	1 次元畳み込みニューラルネットワーク（1-Dimension Convolutional Neural Network）の略です。ニューラルネットワークの手法であり、画像解析に用いられる CNN を、時系列数値データ解析用にカスタマイズした手法です。
1D DCN	1-Dimension Deep Convolutional Neural Network の略です。1D CNN よりもさらに深いネットワークでの高精度な学習を可能にします。
1D OCN	1-Dimension OneClass Neural Network の略です。正常データのみによる学習を可能にします。
カテゴリ	ある基準に基づいてデータをグループ分けした際の個々の区分です。
分類	入力データがどのカテゴリに属するかを出力することです。例えば、センサの時系列数値データから、「正常」か「異常」へ分類します。
回帰	入力データから予測される数値を出力することです。例えば、センサの時系列数値データから、機器の制御値を予測して出力します。
OneClass 分類	正常データのみを学習させることで、正常・異常の分類を行う機能です。
ラベル	学習において、理想の出力を指定するための情報です。分類では、あるデータが属すべきカテゴリです。回帰では、あるデータに対して出力すべき

	数値です。
ラベル付け	時系列数値データの一部に対するラベルを設定する作業です。
ラベルファイル	ラベル付けの結果を保存したファイルです。時系列データのファイル名およびその時間区間と、それに対するラベルが記述されます。
前処理	学習、予測を行う前に、時系列数値データに対して行う処理です。
学習	ラベルファイルを用いて、ニューラルネットワークが、入力した時系列数値データと対になったラベルを出力するために有用なルールを抽出する処理です。
予測モデル	学習の結果、有用なルールを抽出したニューラルネットワークです。
予測	予測モデルを用いて、入力した時系列数値データに対して、分類または回帰を行う処理です。
正規化	データを一定のルールに基づき変換することです。本書では、時系列数値データの値を、特定の範囲内の値に変換する処理を指します。
標準化	データを一定のルールに基づき変換することです。本書では、時系列数値データの値を、その統計情報（平均・標準偏差）を基に、平均が 0、標準偏差が 1 になるよう、データに変換する処理を指します。
サンプリング補正	任意の時系列数値データを基に、指定した周期での時系列数値データを作成する処理です。
ウィンドウサイズ	時系列数値データの一部を順次抜き出して学習・予測に用いる際の、入力として抜き出す時系列数値データの単位（ウィンドウ）のサイズです。
ミニバッチ学習	ニューラルネットワークを学習する手法のひとつで、複数の学習データを一度に学習していく手法です。GPU 上で学習を行う場合は計算の効率が上がるため、処理が高速化されます。
バッチサイズ	一度に学習するデータ件数のことを指します。バッチサイズが大きくなるほど、学習の収束速度が低下する代わりに学習が安定して収束するようになる効果があります。
確信度	予測モデルが、出力した予測結果に対してどの程度強く確信を抱いているかを表した指標です。確信度が高い場合でも、必ずしも予測精度が高いとは限りません。
ソケット	TCP ソケットのことです。ネットワークを介してデータを送受信する際に使用される仕組みです。
GPU	Graphics Processing Unit の略です。3D グラフィックに関する計算処理に特化した半導体チップのことです。並列計算に適しており、この特性を利用することで処理を高速化できます。

第2章 本製品について

2.1 システム構成

本製品のシステム構成を『図 2-1』に記載します。

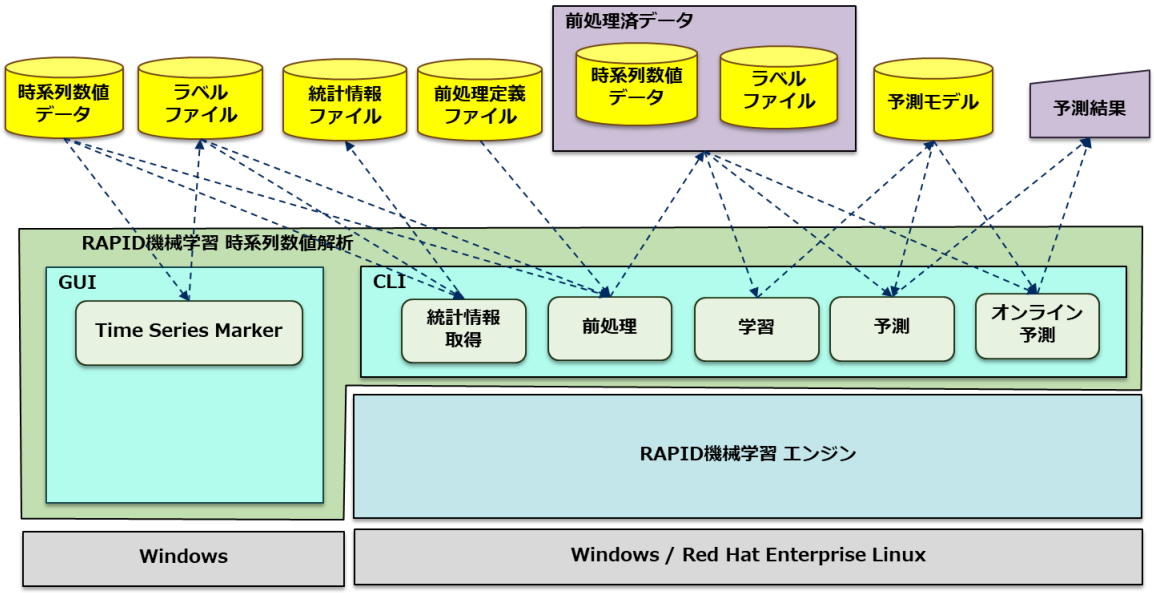

図 2-1 システム構成図

2.2 モジュール概要

『図 2-1』の各モジュールの説明を『表 2-1』に記載します。

表 2-1 モジュール説明

モジュール	説明
Time Series Marker	時系列数値データの学習に必要となるラベルファイル作成を支援します。
統計情報取得	CLI が提供する機能で、時系列数値データの統計情報（平均、標準偏差、最小値、最大値）を計算して出力します。
前処理	CLI が提供する機能で、時系列数値データの正規化やサンプリング処理を行うモジュールです。
学習	CLI が提供する機能で、時系列数値データとラベルファイルを使って学習を行い、予測モデルを作成します。
予測	CLI が提供する機能で、予測モデルを使用し、時系列数値データの分類、回帰、OneClass 分類を行い、その結果を返却します。
オンライン予測	CLI が提供する機能で、予測コマンドを常駐プロセス化することによりプロセスの初期化に必要な時間を短縮し、高速に分析結果を返却します。
RAPID 機械学習 エンジン	本製品の rpd スクリプトや機械学習アルゴリズムの実行基盤となるランタイムです。

第3章 使用方法

3.1 時系列数値データ分析の流れ

本製品を活用した時系列数値データ分析の流れを『図 3-1』に記載します。時系列数値データ分析は、①学習データ作成フェーズ、②学習フェーズ、③予測フェーズの 3 つのフェーズから構成されます。

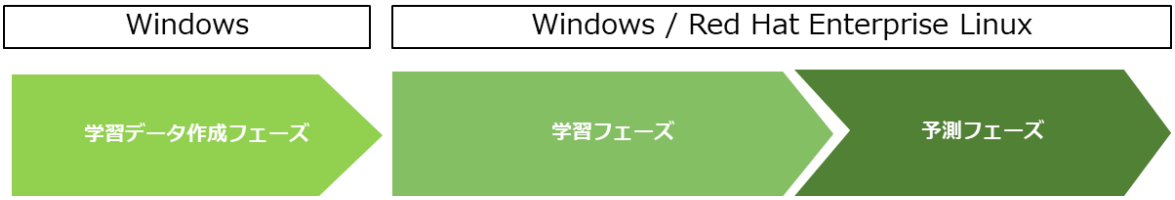

図 3-1 時系列数値データ分析の流れ

3.1.1 学習データ作成フェーズ

学習データ作成フェーズは、Windows 上での作業を想定します。

ラベルファイル作成を Time Series Marker で実施します。Time Series Marker の使用方法については、『Time Series Marker 利用ガイド』を参照してください。

ラベル付けについては、分類、回帰、OneClass 分類によって方針が異なります。『3.2 学習と予測について』を参照してください。

3.1.2 学習フェーズ

学習フェーズは、Windows / Rad Hat Enterprise Linux 上での作業を想定します。

学習データ作成フェーズで作成したラベルファイルについて、前処理機能によって、時系列数値データとラベルファイルを学習に適した状態に処理します。前処理を行ったデータを使って学習機能による学習を行い、予測モデルを作成します。

学習については分類と回帰の 2 つのモードのいずれかを指定します。モードの違いについては、『3.2 学習と予測について』を参照してください。

3.1.3 予測フェーズ

予測フェーズは、Windows / Rad Hat Enterprise Linux 上での作業を想定します。

予測フェーズでは、前処理機能によって、予測に使用する時系列数値データの前処理を行った後で予測を行い、予測結果を取得します。

3.2 学習と予測について

3.2.1 分類

本製品の分類の概要を『図 3-2』に記載します。

分類による学習の場合、ラベルを文字列と判断して予測モデルを作成します。この予測モデルを使用した場合の予測結果は、ラベル付けを行ったラベルごとに確信度が出力されます。

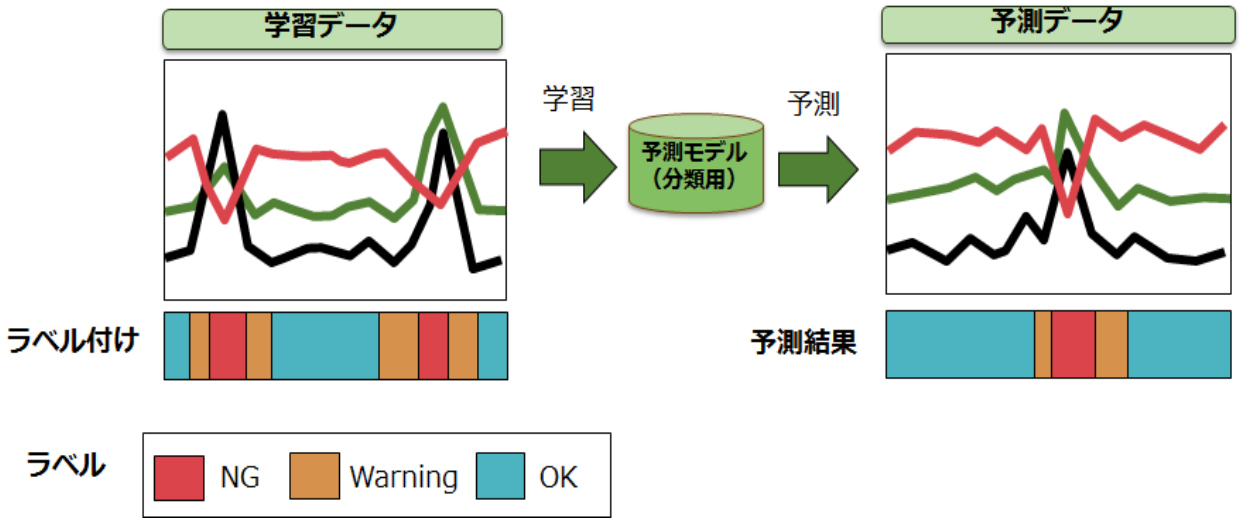

図 3-2 分類の概要図

この場合の予測結果は、ラベル付けを行った、「NG」、「Warning」、「OK」のうち、確信度が最大のものを表示しています。

3.2.2 回帰

本製品の回帰の概要を『図 3-3』に記載します。

回帰による学習の場合、ラベルを数値と判断して予測モデルを作成します。この予測モデルを使用した場合の予測結果は、数値が出力されます。この場合の数値はラベル付けを行った数値以外の値も返されます。

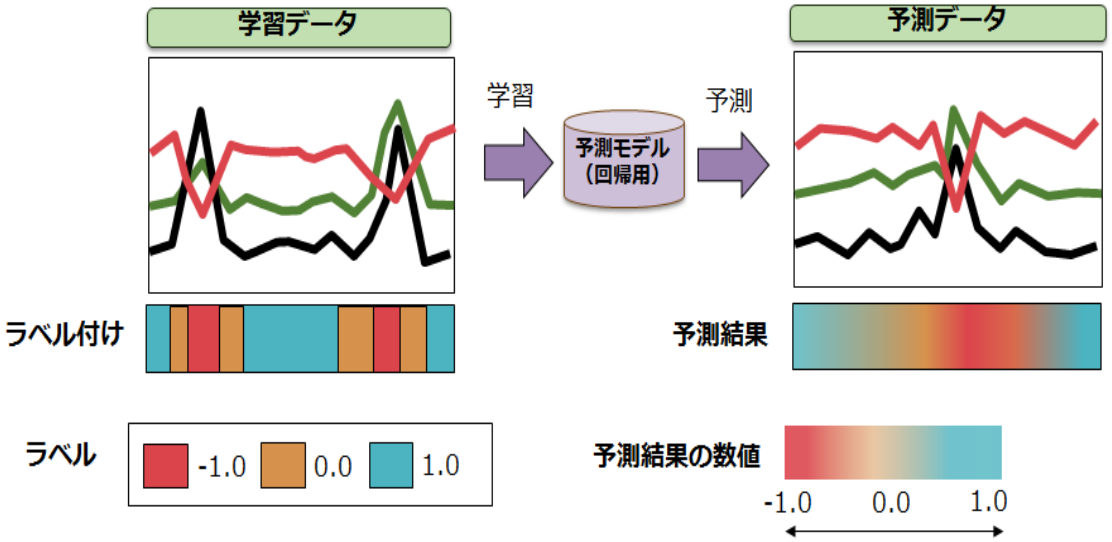

図 3-3 回帰の概要図

『図 3-3』は、『3.2.1 分類』の例にある、「NG」、「Warning」、「OK」の各ラベルの状態を数値で表現した場合を想定したもので、それぞれ、「-1.0」、「0.0」、「1.0」の 3 つをラベル付けします。ています。ラベル「0.0」を基準として、安全と考えられる状態をラベル「1.0」、危険と考えられる状態をラベル「-1.0」として考えています。

この場合の予測結果の例としては、-1.0 から 1.0 までを中心とした値が出力されます。-1.0 に近い値ほどより危険な状態と予測され、1.0 に近い値ほどより安全な状態という意味となります。

3.2.3 OneClass 分類

本製品の OneClass 分類の概要を『図 3-4』に記載します。

OneClass 分類では、正常ラベルを付けたデータのみを使用して学習し、予測モデルを作成します。予測モデルを作成する場合は、特徴抽出区間の時系列数値データを入力したとき、それに続く状態判断区間の時系列数値データが正常であると判断する予測モデルを作成します。

この予測モデルを使用した場合の予測結果は、各予測区間の異常可能性を示す値として異常度が出力され、その値が設定値（許容度）を超えた箇所が異常として検出され、それ以外の箇所は正常と判断されます。

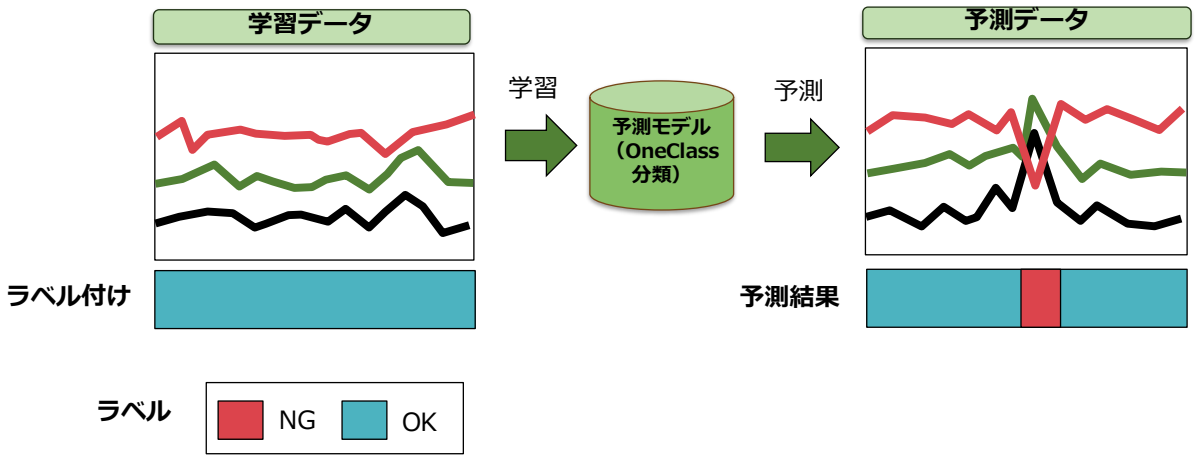

図 3-4 OneClass 分類の概要図

⚠注意 OneClass 分類のラベル指定方法について

・OneClass 分類のラベルは正常「OK」、異常「NG」の 2 値で固定となっています。

第4章 CLI 使用例

本章では、CLI の代表的な使用例について説明します。
主に Rad Hat Enterprise Linux を想定して記載しているため、Windows で実行する場合は、使用するファイルのパスを適宜読み替えてください。

4.1 分類の使用例

ある部品に関連するセンサのデータをもとに故障を検知するケースを想定します。センサデータからの時系列数値データをもとに予測し、OK（正常）、Warning（警告）、NG（異常）の各状態を分類することを想定します。センサデータは決まった時刻間隔（ここでは 1 分）で取得するものとします。

4.1.1 学習データ生成フェーズ

時系列数値データ作成

本作業では CLI を使用しませんが、CLI 実施の入力データとなるため、説明します。

時系列数値データのフォーマットに従った時系列数値データを用意します。

時系列数値データの例として、以下の内容の”train_data_cls.csv”を作成します。

```
[Header]"time","sensor1","sensor2"
0:01:00,10,0.3
0:02:00,9,0.45
0:03:00,8,0.4
0:04:00,7,0.43
0:05:00,6,0.6
0:06:00,5,0.3
0:07:00,5,0.34
0:08:00,2,0.42
0:09:00,1,0.51
```

ラベルファイル作成

本作業では CLI を使用しませんが、CLI 実施の入力データとなるため、説明します。

『Time Series Marker 利用ガイド』の手順に従って、ラベルファイルを作成します。

異常となった区間を「NG」、異常と疑われる警告に相当する区間を「Warning」、正常に動作して

いる区間を「OK」としてラベル付けします。

時系列数値データファイルと、それに対して作成したラベルファイルは、必ず同じディレクトリに保存してください。

作成したラベルファイルの例として、以下の内容の「train_data_cls.lab」を作成します。

```
[DataPath]"train_data_cls.csv"
0:01:00,0:03:00,"OK"
0:04:00,0:06:00,"Warning"
0:08:00,0:09:00,"NG"
```

4.1.2 学習フェーズ

前処理

1. 前処理定義ファイルの作成

時系列数値データを処理するための前処理定義ファイルを作成します。前処理定義ファイルの作成に関しては『6.4 前処理定義ファイル』を参照してください。

前処理定義ファイルの例として、以下の内容の「/home/user/method/method_file.json」を作成します。

時系列数値データのサンプリング補正を 0.1 秒周期で行ったうえで、正規化を行って区間を[-1.0,1.0]に揃える前処理を定義しています。

```
{
    "unity":
    [
        {
            "method":"sampling",
            "param":
            {
                "type":"linear",
                "rate":0.1
            }
        },
        {
            "method":"normalize",
            "param":
            [
                {
```

```
                    "target":0,
                    "src_range":[0,10],
                    "dst_range":[-1.0,1.0]
                }
            ]
        }
    ]
}
```

2. 前処理コマンドの実施

学習データ生成フェーズで作成したデータ「train_data_cls.csv」、「train_data_cls.lab」は、Red Hat Enterprise Linux サーバ上で参照できる場所へ移動してください。ここでは、「/home/user/data/preprocess」へデータを格納したものとします。

前処理コマンド『exec_preprocess』を使用し、時系列数値データの前処理を行います。

ここでは、時系列数値データに紐づくラベルファイルがあるため、-l を指定します。ラベルファイル「train_data_cls.lab」には、同じディレクトリに存在する時系列数値データファイル「train_data_cls.csv」に対するラベルが記載されているものとします。

前処理コマンドの実施例は以下となります。

```
rapid /S TSA/exec_preprocess.rpd -pre /home/user/method/method_file.json -l /home/user/data/preprocess/train_data_cls.lab -suf prep
```

上記の前処理コマンドを実行すると、ディレクトリ「/home/user/data/preprocess」に、前処理済み時系列数値データファイル「train_data_cls_prep.csv」とそれに対する新しいラベルファイル「train_data_cls_prep.lab」が出力されます。

学習

1. 学習の実施

前処理コマンド実行の結果、出力された前処理済み時系列数値データファイル「train_data_cls_prep.csv」とラベルファイル「train_data_cls_prep.lab」を別ディレクトリに格納します。

特に、ディレクトリを指定して学習を行う場合は、前処理の結果出力されたファイルを必ず別ディレクトリに移動してから、実行してください。

ここでは学習に使用する時系列数値データファイルとラベルファイルを「/home/user/data/train」へ格納したものとします。

学習コマンド『exec_train』を使用して学習を実行し、予測モデルを作成します。

ここでは分類用の予測モデルを作成するため、-mode オプションに「cls」を指定します。

学習アルゴリズム-t には「1DCNN」、「1DDCN」が指定できますが、ここでは、「1DCNN」を指定します。

また、学習コンフィグファイルとして、「/home/user/conf/tsa_train.conf」にデフォルトのコンフィグファイルをコピーしたものとします。

```
cp /opt/nec/rapid/etc/TSA/tsa_train.conf /home/user/conf/tsa_train.conf
```

学習コマンドの実施例は以下となります。

```
rapid /S TSA/exec_train.rpd -t 1DCNN -mode cls -l /home/user/data/train/train_data_cls_prep.lab -m /home/user/model/train_cls -param /home/user/conf/tsa_train.conf
```

上記の学習コマンドを実行すると、ディレクトリ「/home/user/model/train_cls」に、学習済みの予測モデルが出力されます。

4.1.3 予測フェーズ

時系列数値データ作成

本作業では CLI を使用しませんが、CLI 実施の入力データとなるため、説明します。

時系列数値データのフォーマットに従った時系列数値データを用意します。

時系列数値データの例として、以下の内容の”predict_data_cls.csv”を作成します。

```
[Header]"time","sensor1","sensor2"
0:01:00,9,0.4
0:02:00,8,0.55
0:03:00,7,0.5
0:04:00,6,0.53
0:05:00,5,0.7
0:06:00,4,0.4
0:07:00,4,0.44
```

```
0:08:00,1,0.52
0:09:00,0,0.61
```

前処理

1. 前処理定義ファイルの作成

前処理定義ファイルを作成します。

ここでは、学習フェーズで使用した前処理定義ファイル「/home/user/method/method_file.json」を使用するものとします。

2. 前処理

予測を行う時系列数値データの前処理を行います。

```
rapid /S TSA/exec_preprocess.rpd -pre /home/user/method/method_file.json -i /home/user/data/preprocess/predict_data_cls.csv -suf prep
```

上記コマンドを実行すると、ディレクトリ「/home/user/data/preprocess」に、前処理済み時系列数値データファイル「predict_data_cls_prep.csv」が出力されます。

予測

1. 予測の実施

前処理コマンド実行の結果、出力された前処理済み時系列数値データファイル「predict_data_cls_prep.csv」を別ディレクトリに格納します。

特に、ディレクトリを指定して予測を行う場合は、前処理の結果出力されたファイルを必ず別ディレクトリに移動してから、実行してください。

ここでは予測に使用する時系列数値データファイルを「/home/user/data/predict」へ格納したものとします。

予測コマンド『exec_predict』を使用して時系列数値データの予測を実行します。

ここでは予測コンフィグファイルとして、デフォルトの予測コンフィグファイル「/opt/nec/rapid/etc/TSA/tsa_predict.conf」を「/home/user/conf/tsa_predict.conf」へコピーしたものとします。

```
cp /opt/nec/rapid/etc/TSA/tsa_predict.conf /home/user/conf/tsa_predict.conf
```

予測コマンドの実施例は以下となります。

```
rapid /S TSA/exec_predict.rpd -i /home/user/data/predict/predict_data_cls_prep.csv -m /home/user/model/train_cls -param /home/user/conf/tsa_predict.conf
```

上記コマンドを実行すると、ディレクトリ「/home/user/data/predict」に、簡易予測結果ファイル「predict_data_cls_prep.result」と詳細予測結果ファイル「predict_data_cls_prep.dresult」が出力されます。

簡易予測結果ファイル「predict_data_cls_prep.result」の出力例は、以下となります。

予測結果ファイルの詳細は『6.3 結果ファイル』を参照してください。

```
[DataPath]"predict_data_cls_prep.csv"
0:01:00.000000,0:01:06.300000,"OK"
0:01:00.100000,0:01:06.400000,"OK"
0:01:00.200000,0:01:06.500000,"OK"
…
0:02:29.600000,0:02:35.900000,"OK"
0:02:29.700000,0:02:36.000000,"OK"
0:02:29.800000,0:02:36.100000,"Warning"
0:02:29.900000,0:02:36.200000,"Warning"
…
0:07:03.400000,0:07:09.700000,"Warning"
0:07:03.500000,0:07:09.800000,"Warning"
0:07:03.600000,0:07:09.900000,"NG"
0:07:03.700000,0:07:10.000000,"NG"
```

詳細予測結果ファイル「predict_data_cls_prep.dresult」の出力例は、以下となります。

[DataPath]"predict_data_cls.csv"

0:01:00.000000,0:01:06.300000,"OK",0.99998426437378,"Warning",8.9745735749602e-06,"NG",6.7569808379631e-06

0:01:00.100000,0:01:06.400000,"OK",0.99998426437378,"Warning",8.9724253484746e-06,"NG",6.7620862864715e-06

0:01:00.200000,0:01:06.500000,"OK",0.99998426437378,"Warning",8.9702607510844e-06,"NG",6.7671053329832e-06

…

0:02:29.600000,0:02:35.900000,"OK",0.50656867027283,"Warning",0.49242335557938,"NG",0.0010079514468089

0:02:29.700000,0:02:36.000000,"OK",0.50102859735489,"Warning",0.49796360731125,"NG",0.0010077991755679

0:02:29.800000,0:02:36.100000,"Warning",0.50350338220596,"OK",0.49548909068108,"NG",0.0010075230384246

0:02:29.900000,0:02:36.200000,"Warning",0.50904220342636,"OK",0.48995071649551,"NG",0.0010071243159473

…

0:07:03.400000,0:07:09.700000,"Warning",0.51309764385223,"NG",0.4860677421093,"OK",0.00083460472524166

0:07:03.500000,0:07:09.800000,"Warning",0.50619798898697,"NG",0.49296629428864,"OK",0.00083572423318401

0:07:03.600000,0:07:09.900000,"NG",0.49986633658409,"Warning",0.4992969930172,"OK",0.00083668390288949

0:07:03.700000,0:07:10.000000,"NG",0.50676590204239,"Warning",0.49239671230316,"OK",0.00083749007899314

4.2 回帰の使用例

ある部品に関連するセンサのデータをもとに故障を検知するケースを想定します。センサデータからの時系列数値データに、ラベル「0.0」を基準として、安全と考えられる状態をラベル「1.0」、危険と考えられる状態をラベル「-1.0」としてラベル付けを行います。
センサデータは決まった時刻間隔（ここでは 1 分）で取得するものとします。

この場合の予測結果の例としては、-1.0 から 1.0 までを中心とした値が出力されます。-1.0 に近い値ほどより危険な状態と予測され、1.0 に近い値ほどより安全な状態という意味となります。

4.2.1 学習データ生成フェーズ

時系列数値データ作成

本作業では CLI を使用しませんが、CLI 実施の入力データとなるため、説明します。

時系列数値データのフォーマットに従った時系列数値データを用意します。

時系列数値データの例として、以下の内容の”train_data_reg.csv”を作成します。

```
[Header]"time","sensor1","sensor2"
0:01:00,10,0.3
0:02:00,9,0.45
0:03:00,8,0.4
0:04:00,7,0.43
0:05:00,6,0.6
0:06:00,5,0.3
0:07:00,5,0.34
0:08:00,2,0.42
0:09:00,1,0.51
```

ラベルファイル作成

本作業では CLI を使用しませんが、CLI 実施の入力データとなるため、説明します。

『Time Series Marker 利用ガイド』の手順に従って、ラベルファイルを作成します。

異常となった区間を「-1.0」、異常と疑われる警告に相当する区間を「0.0」、正常に動作している区間を「1.0」としてラベル付けします。

時系列数値データファイルと、それに対して作成したラベルファイルは、必ず同じディレクトリに保存してください。

作成したラベルファイルの例として、以下の内容の「train_data_reg.lab」を作成します。

```
[DataPath]"train_data_reg.csv"
0:01:00,0:03:00,"1.0"
0:04:00,0:06:00,"0.0"
0:08:00,0:09:00,"-1.0"
```

4.2.2 学習フェーズ

前処理

1. 前処理定義ファイルの作成

時系列数値データを処理するための前処理定義ファイルを作成します。前処理定義ファイルの作成に関しては『6.4 前処理定義ファイル』を参照してください。

前処理定義ファイルの例として、以下の内容の「/home/user/method/method_file.json」を作成します。

時系列数値データのサンプリング補正を 0.1 秒周期で行ったうえで、正規化を行って区間を[-1.0,1.0]に揃える前処理を定義しています。

```
{
    "unity":
    [
        {
            "method":"sampling",
            "param":
            {
                "type":"linear",
                "rate":0.1
            }
        },
        {
            "method":"normalize",
            "param":
            [
                {
                    "target":0,
                    "src_range":[0,10],
                    "dst_range":[-1.0,1.0]
                }
            ]
        }
    ]
}
```

2. 前処理コマンドの実施

学習データ生成フェーズで作成したデータは、Red Hat Enterprise Linux サーバ上で参照できる場所へ移動してください。ここでは、「/home/user/data/preprocess」へデータを格納したものとします。

前処理コマンド『exec_preprocess』を使用し、時系列数値データの前処理を行います。

ここでは、時系列数値データに紐づくラベルファイルがあるため、-l を指定します。ラベルファイル「train_data_reg.lab」には、同じディレクトリに存在する時系列数値データファイル「train_data_reg.csv」に対するラベルが記載されているものとします。

前処理コマンドの実施例は以下となります。

```
rapid /S TSA/exec_preprocess.rpd -pre /home/user/method/method_file.json -l /home/user/data/preprocess/train_data_reg.lab -suf prep
```

上記の前処理コマンドを実行すると、ディレクトリ「/home/user/data/preprocess」に、前処理済み時系列数値データファイル「train_data_reg_prep.csv」とそれに対する新しいラベルファイル「train_data_reg_prep.lab」が出力されます。

学習

1. 学習の実施

前処理コマンド実行の結果、出力された前処理済み時系列数値データファイル「train_data_reg_prep.csv」とラベルファイル「train_data_reg_prep.lab」を別ディレクトリに格納します。

特に、ディレクトリを指定して学習を行う場合は、前処理の結果出力されたファイルを必ず別ディレクトリに移動してから、実行してください。

ここでは学習に使用する時系列数値データファイルとラベルファイルを「/home/user/data/train」へ格納したものとします。

学習コマンド『exec_train』を使用して学習を実行し、予測モデルを作成します。
ここでは回帰用の予測モデルを作成するため、-mode オプションに「reg」を指定します。
学習アルゴリズム-t には「1DCNN」、「1DDCN」が指定できますが、ここでは、「1DCNN」を指定します。

また、学習コンフィグファイルとして、「/home/user/conf/tsa_train.conf」にデフォルトのコンフィグファイルをコピーしたものとします。

```
cp /opt/nec/rapid/etc/TSA/tsa_train.conf /home/user/conf/tsa_train.conf
```

学習コマンドの実施例は以下となります。

```
rapid /S TSA/exec_train.rpd -t 1DCNN -mode reg -l /home/user/data/train/train_data_reg_prep.lab -m /home/user/model/train_reg -param /home/user/conf/tsa_train.conf
```

上記の学習コマンドを実行すると、ディレクトリ「/home/user/model/train_reg」に、学習済みの予測モデルが出力されます。

4.2.3 予測フェーズ

時系列数値データ作成

本作業では CLI を使用しませんが、CLI 実施の入力データとなるため、説明します。

時系列数値データのフォーマットに従った時系列数値データを用意します。

時系列数値データの例として、以下の内容の”predict_data_reg.csv”を作成します。

```
[Header]"time","sensor1","sensor2"
0:01:00,9,0.4
0:02:00,8,0.55
0:03:00,7,0.5
0:04:00,6,0.53
0:05:00,5,0.7
0:06:00,4,0.4
0:07:00,4,0.44
0:08:00,1,0.52
0:09:00,0,0.61
```

前処理

1. 前処理定義ファイルの作成

前処理定義ファイルを作成します。

ここでは、学習フェーズで使用した前処理定義ファイル「/home/user/method/method_file.json」を使用するものとします。

2. 前処理

予測を行う時系列数値データの前処理を行います。

```
rapid /S TSA/exec_preprocess.rpd -pre /home/user/method/method_file.json -i /home/user/data/preprocess/predict_data_reg.csv -suf prep
```

上記コマンドを実行すると、ディレクトリ「/home/user/data/preprocess」に、前処理済み時系列数値データファイル「predict_data_reg_prep.csv」が出力されます。

予測

2. 予測の実施

前処理コマンド実行の結果、出力された前処理済み時系列数値データファイル「predict_data_reg_prep.csv」を別ディレクトリに格納します。

特に、ディレクトリを指定して予測を行う場合は、前処理の結果出力されたファイルを必ず別ディレクトリに移動してから、実行してください。

ここでは予測に使用する時系列数値データファイルを「/home/user/data/predict」へ格納したものとします。

予測コマンド『exec_predict』を使用して時系列数値データの予測を実行します。

ここでは予測コンフィグファイルとして、デフォルトの予測コンフィグファイル「/opt/nec/rapid/etc/TSA/tsa_predict.conf」を「/home/user/conf/tsa_predict.conf」へコピーしたものとします。

```
cp /opt/nec/rapid/etc/TSA/tsa_predict.conf /home/user/conf/tsa_predict.conf
```

予測コマンドの実施例は以下となります。

```
rapid /S TSA/exec_predict.rpd -i /home/user/data/predict/predict_data_reg_prep.csv -m /home/user/model/train_reg -param /home/user/conf/tsa_predict.conf
```

上記コマンドを実行すると、ディレクトリ「/home/user/data/predict」に、簡易予測結果ファイル「predict_data_reg_prep.result」が出力されます。

簡易予測結果ファイル「predict_data_reg_prep.result」の出力例は、以下となります。

予測結果ファイルの詳細は『6.3 結果ファイル』を参照してください。

```
[DataPath]"predict_data_reg_prep.csv"
0:01:00.000000,0:01:06.300000,"1.0188482999802"
0:01:00.100000,0:01:06.400000,"1.0188522338867"
0:01:00.200000,0:01:06.500000,"1.0188554525375"
0:01:00.300000,0:01:06.600000,"1.0188587903976"
…
0:04:10.000000,0:04:16.300000,"0.00012225061072968"
0:04:10.100000,0:04:16.400000,"0.00010099069913849"
0:04:10.200000,0:04:16.500000,"7.9898454714566e-05"
0:04:10.300000,0:04:16.600000,"5.8686880947789e-05"
…
0:07:25.500000,0:07:31.800000,"-0.998959004879"
0:07:25.600000,0:07:31.900000,"-0.99991142749786"
0:07:25.700000,0:07:32.000000,"-1.0008531808853"
0:07:25.800000,0:07:32.100000,"-1.0017877817154"
```

4.3 OneClass 分類の使用例

ある部品に関連するセンサのデータをもとに故障を検知するケースを想定します。センサデータからの時系列数値データの正常区間を学習し、正常（OK）、異常（NG）の各状態を分類することを想定します。センサデータは決まった時刻間隔（ここでは 1 分）で取得するものとします。

4.3.1 学習データ生成フェーズ

時系列数値データ作成

本作業では CLI を使用しませんが、CLI 実施の入力データとなるため、説明します。

時系列数値データのフォーマットに従った時系列数値データを用意します。

時系列数値データの例として、以下の内容の”train_data_1cls.csv”を作成します。

```
[Header]"time","sensor1","sensor2"
0:01:00,10,0.3
0:02:00,9,0.45
0:03:00,8,0.4
0:04:00,7,0.43
0:05:00,6,0.6
0:06:00,5,0.3
0:07:00,5,0.34
0:08:00,2,0.42
0:09:00,1,0.51
```

ラベルファイル作成

本作業では CLI を使用しませんが、CLI 実施の入力データとなるため、説明します。

『Time Series Marker 利用ガイド』の手順に従って、ラベルファイルを作成します。

正常に動作している区間のみを「OK」としてラベル付けします。

時系列数値データファイルと、それに対して作成したラベルファイルは、必ず同じディレクトリに保存してください。

作成したラベルファイルの例として、以下の内容の「train_data_1cls.lab」を作成します。

```
[DataPath]"train_data_1cls.csv"
0:01:00,0:03:00,"OK"
```

4.3.2 学習フェーズ

前処理

1. 前処理定義ファイルの作成

時系列数値データを処理するための前処理定義ファイルを作成します。前処理定義ファイルの作成に関しては『6.4 前処理定義ファイル』を参照してください。

前処理定義ファイルの例として、以下の内容の「/home/user/method/method_file.json」を作成します。

時系列数値データのサンプリング補正を 0.1 秒周期で行ったうえで、正規化を行って区間を[-1.0,1.0]に揃える前処理を定義しています。

```
{
    "unity":
    [
        {
            "method":"sampling",
            "param":
            {
                "type":"linear",
                "rate":0.1
            }
        },
        {
            "method":"normalize",
            "param":
            [
                {
                    "target":0,
                    "src_range":[0,10],
                    "dst_range":[-1.0,1.0]
                }
            ]
        }
    ]
}
```

2. 前処理コマンドの実施

学習データ生成フェーズで作成したデータは、Red Hat Enterprise Linux サーバ上で参照できる場所へ移動してください。ここでは、「/home/user/data/preprocess」へデータを格納したものとします。

前処理コマンド『exec_preprocess』を使用し、時系列数値データの前処理を行います。

ここでは、時系列数値データに紐づくラベルファイルがあるため、-l を指定します。ラベルファイル「train_data_1cls.lab」には、同じディレクトリに存在する時系列数値データファイル「train_data_1cls.csv」に対するラベルが記載されているものとします。

また、ここでは、前処理定義ファイルとして、「/home/user/method/method_file.json」を作成したものとします。

前処理コマンドの実施例は以下となります。

```
rapid /S TSA/exec_preprocess.rpd -pre /home/user/method/method_file.json -l /home/user/data/preprocess/train_data_1cls.lab -suf prep
```

上記の前処理コマンドを実行すると、ディレクトリ「/home/user/data/preprocess」に、前処理済み時系列数値データファイル「train_data_1cls_prep.csv」とそれに対する新しいラベルファイル「train_data_1cls_prep.lab」が出力されます。

学習

1. 学習の実施

前処理コマンド実行の結果、出力された前処理済み時系列数値データファイルとラベルファイルを別ディレクトリに格納します。

特に、ディレクトリを指定して学習を行う場合は、前処理の結果出力されたファイルを必ず別ディレクトリに移動してから、実行してください。

ここでは学習に使用する時系列数値データファイルとラベルファイルを「/home/user/data/train」へ格納したものとします。

学習コマンド『exec_train』を使用して学習を実行し、予測モデルを作成します。

ここでは分類用の予測モデルを作成するため、-mode オプションに「cls」を指定します。
学習アルゴリズム-t には「1DOCN」を指定します。

また、学習コンフィグファイルとして、「/home/user/conf/tsa_train.conf」にデフォルトのコンフィグファイルをコピーしたものとします。

```
cp /opt/nec/rapid/etc/TSA/tsa_train.conf /home/user/conf/tsa_train.conf
```

学習コマンドの実施例は以下となります。

```
rapid /S TSA/exec_train.rpd -t 1DOCN -mode cls -l /home/user/data/train/train_data_1cls_prep.lab -m /home/user/model/train_1cls -param /home/user/conf/tsa_train.conf
```

上記の学習コマンドを実行すると、ディレクトリ「/home/user/model/train_1cls」に、学習済みの予測モデルが出力されます。

4.3.3 予測フェーズ

時系列数値データ作成

本作業では CLI を使用しませんが、CLI 実施の入力データとなるため、説明します。

時系列数値データのフォーマットに従った時系列数値データを用意します。

時系列数値データの例として、以下の内容の”predict_data_1cls.csv”を作成します。

```
[Header]"time","sensor1","sensor2"
0:01:00,9,0.4
0:02:00,8,0.55
0:03:00,7,0.5
0:04:00,6,0.53
0:05:00,5,0.7
0:06:00,4,0.4
0:07:00,4,0.44
0:08:00,1,0.52
0:09:00,0,0.61
```

前処理

1. **前処理定義ファイルの作成**

前処理定義ファイルを作成します。

ここでは、学習フェーズで使用した前処理定義ファイル「/home/user/method/method_file.json」を使用するものとします。

2. 前処理

予測を行う時系列数値データの前処理を行います。

```
rapid /S TSA/exec_preprocess.rpd -pre /home/user/method/method_file.json -i /home/user/data/preprocess/predict_data_1cls.csv -suf prep
```

上記コマンドを実行すると、ディレクトリ「/home/user/data/preprocess」に、前処理済み時系列数値データファイル「predict_data_1cls_prep.csv」が出力されます。

予測

1. 予測の実施

前処理コマンド実行の結果、出力された前処理済み時系列数値データファイルを別ディレクトリに格納します。

特に、ディレクトリを指定して予測を行う場合は、前処理の結果出力されたファイルを必ず別ディレクトリに移動してから、実行してください。

ここでは予測に使用する時系列数値データファイルを「/home/user/data/predict」へ格納したものとします。

予測コマンド『exec_predict』を使用して時系列数値データの予測を実行します。

ここでは予測コンフィグファイルとして、デフォルトの予測コンフィグファイル「/opt/nec/rapid/etc/TSA/tsa_predict.conf」を「/home/user/conf/tsa_predict.conf」へコピーしたものとします。

```
cp /opt/nec/rapid/etc/TSA/tsa_predict.conf /home/user/conf/tsa_predict.conf
```

予測コマンドの実施例は以下となります。

```
rapid /S TSA/exec_predict.rpd -i /home/user/data/predict/predict_data_1cls_prep.csv -m /home/user/model/train_1cls -param /home/user/conf/tsa_predict.conf
```

上記コマンドを実行すると、ディレクトリ「/home/user/data/predict」に、簡易予測結果ファイル「predict_data_1cls_prep.result」と詳細予測結果ファイル「predict_data_1cls_prep.dresult」が出力されます。

簡易予測結果ファイル「predict_data_cls_prep.result」の出力例は、以下となります。

予測結果ファイルの詳細は『6.3 結果ファイル』を参照してください。

```
[DataPath]"predict_data_1cls_prep.csv"
0:01:00.000000,0:01:07.100000,"NG"
0:01:00.100000,0:01:07.200000,"NG"
0:01:00.200000,0:01:07.300000,"NG"
…
0:02:29.600000,0:02:36.700000,"NG"
0:02:29.700000,0:02:36.800000,"NG"
0:02:29.800000,0:02:36.900000,"NG"
0:02:29.900000,0:02:37.000000,"NG"
…
0:07:03.400000,0:07:10.500000,"NG"
0:07:03.500000,0:07:10.600000,"NG"
0:07:03.600000,0:07:10.700000,"NG"
0:07:03.700000,0:07:10.800000,"NG"
```

詳細予測結果ファイル「predict_data_cls_prep.dresult」の出力例は、以下となります。

```
[DataPath]"predict_data_1cls_prep.csv"
0:01:00.000000,0:01:07.100000,"NG",3.2010872364044,"OK",1
0:01:00.100000,0:01:07.200000,"NG",3.1794202327728,"OK",1
0:01:00.200000,0:01:07.300000,"NG",3.1536500453949,"OK",1
…
0:02:29.600000,0:02:36.700000,"NG",20.765460968018,"OK",1
0:02:29.700000,0:02:36.800000,"NG",20.815092086792,"OK",1
0:02:29.800000,0:02:36.900000,"NG",20.864700317383,"OK",1
0:02:29.900000,0:02:37.000000,"NG",20.919801712036,"OK",1
…
0:07:03.400000,0:07:10.500000,"NG",220.00918579102,"OK",1
0:07:03.500000,0:07:10.600000,"NG",220.28395080566,"OK",1
0:07:03.600000,0:07:10.700000,"NG",220.55862426758,"OK",1
0:07:03.700000,0:07:10.800000,"NG",220.8334197998,"OK",1
```

この結果では、全区間が異常と判定されていますが、少量の異常データがあれば、予測コンフィグファイルにおける、異常度に対する許容度を変更することにより、予測結果を正しく出力できるようになります。例えば、上記の予測対象の時系列数値データについて、最初の 2 分間は正常であり、その後は異常であったということが分かっている場合、「/home/user/conf/tsa_predict.conf」の設定項目のうち、[1DOCN_TOLERANCE]を編集します。

⚠注意　予測コンフィグファイルの[1DOCN_TOLERANCE]項目について

・OneClass 分類の異常判定における異常度に対する許容度を設定します。OneClass 分類の異常判定において、異常度がこの値を超えた場合に NG と判定します。
デフォルトでは 1 となっているため、OneClass 分類をしたいデータや予測モデルに合わせて変更する必要があります。

以下のコマンドを実行し、「/home/user/conf/tsa_predict.conf」を編集します。

```
vi /home/user/conf/tsa_predict.conf
```

【変更前】

```
# common
[SLIDE_SIZE]=1


# 1DOCN
[1DOCN_TOLERANCE]=1.0
[1DOCN_FILTER_SIZE]=1
```

【変更後】

```
# common
[SLIDE_SIZE]=1


# 1DOCN
[1DOCN_TOLERANCE]=10.0
[1DOCN_FILTER_SIZE]=1
```

変更した設定で、再度予測を実行します。

予測結果ファイルの名前を変更するため、-suf を指定します。

```
rapid /S TSA/exec_predict.rpd -i /home/user/data/predict/predict_data_1cls_prep.csv -m
/home/user/model/train_1cls -param /home/user/conf/tsa_predict.conf –suf tolerance_10
```

簡易予測結果ファイル「predict_data_cls_prep_tolerance_10.result」の出力に異常度の閾値が反映されます。

```
[DataPath]"predict_data_1cls_prep.csv"
0:01:00.000000,0:01:07.100000,"OK"
0:01:00.100000,0:01:07.200000,"OK"
0:01:00.200000,0:01:07.300000,"OK"
…
0:02:29.600000,0:02:36.700000,"NG"
0:02:29.700000,0:02:36.800000,"NG"
0:02:29.800000,0:02:36.900000,"NG"
0:02:29.900000,0:02:37.000000,"NG"
…
0:07:03.400000,0:07:10.500000,"NG"
0:07:03.500000,0:07:10.600000,"NG"
0:07:03.600000,0:07:10.700000,"NG"
0:07:03.700000,0:07:10.800000,"NG"
```

異常度の許容度を変更後の詳細予測結果ファイル「predict_data_cls_prep_tolerance_10.dresult」の出力例は、以下となります。

```
[DataPath]"predict_data_1cls_prep.csv"
0:01:00.000000,0:01:07.100000,"OK",10,"NG",3.2010872364044
0:01:00.100000,0:01:07.200000,"OK",10,"NG",3.1794202327728
0:01:00.200000,0:01:07.300000,"OK",10,"NG",3.1536500453949
…
0:02:29.600000,0:02:36.700000,"NG",20.765460968018,"OK",10
0:02:29.700000,0:02:36.800000,"NG",20.815092086792,"OK",10
0:02:29.800000,0:02:36.900000,"NG",20.864700317383,"OK",10
0:02:29.900000,0:02:37.000000,"NG",20.919801712036,"OK",10
…
```

```
0:07:03.400000,0:07:10.500000,"NG",220.00918579102,"OK",10
0:07:03.500000,0:07:10.600000,"NG",220.28395080566,"OK",10
0:07:03.600000,0:07:10.700000,"NG",220.55862426758,"OK",10
0:07:03.700000,0:07:10.800000,"NG",220.8334197998,"OK",10
```

この結果では、最初の２分間は正常、その後は異常と判定されており、予測結果が正しく出力されています。このように、少量の異常データを用いて異常度に対する許容度を変更することにより、これ以外の未知の時系列数値データに対しても、より正しく予測結果を出力できるようになります。

4.4 予測のみ実施する場合

実際の分析作業では、別の環境で作成した予測モデルを移行し、予測のみを実施することもあります。
ここでは、Linux の環境で予測モデルを作成し、Windows の環境で予測をすることを例にします。

4.4.1 予測モデルの移行

『4.1.2 学習フェーズ』で作成した予測モデルを例として説明します。
学習時に使用した前処理定義ファイル、学習時に-m で指定したディレクトリをコピーしてください。

ファイルパス	説明
/home/user/method/method_file.json	学習時に使用した前処理定義ファイル
/home/user/model/train_cls	学習によって作成された予測モデルディレクトリ

これらのファイルを、予測を実行する Windows の環境にコピーします。
ここでは「C:¥method」フォルダ、「C:¥model」フォルダを作成し、その中に前処理定義ファイル、予測モデルディレクトリを配置します。

ファイルパス	説明
C:¥method¥method_file.json	学習時に使用した前処理定義ファイル
C:¥model¥train_cls	学習によって作成された予測モデルディレクトリ

4.4.2 予測フェーズ

時系列数値データ作成

本作業では CLI を使用しませんが、CLI 実施の入力データとなるため、説明します。

時系列数値データのフォーマットに従った時系列数値データを用意します。

時系列数値データの例として、「C:¥data¥preprocess」に、以下の内容の”predict_data_cls.csv”を作成します。

```
[Header]"time","sensor1","sensor2"
0:01:00,9,0.4
0:02:00,8,0.55
0:03:00,7,0.5
0:04:00,6,0.53
0:05:00,5,0.7
0:06:00,4,0.4
0:07:00,4,0.44
0:08:00,1,0.52
0:09:00,0,0.61
```

前処理

予測を行う時系列数値データの前処理を行います。

ここでは、学習フェーズで使用した前処理定義ファイル「C:¥method¥method_file.json」を使用するものとします。

```
rapid /S TSA/exec_preprocess.rpd -pre C:¥method¥method_file.json -i C:¥data¥preprocess¥predict_data_cls.csv -suf prep
```

上記コマンドを実行すると、「C:¥data¥preprocess」フォルダに、前処理済み時系列数値データファイル「predict_data_cls_prep.csv」が出力されます。

予測

1. 予測の実施

前処理コマンド実行の結果、出力された前処理済み時系列数値データファイルを別ディレクトリに格納します。

特に、ディレクトリを指定して予測を行う場合は、前処理の結果出力されたファイルを必ず別ディレクトリに移動してから、実行してください。

ここでは予測に使用する時系列数値データファイルを「C:¥data¥predict」へ格納したものとします。

予測コマンド『exec_predict』を使用して時系列数値データの予測を実行します。

ここでは予測コンフィグファイルとして、デフォルトの予測コンフィグファイル「 C:¥Users¥Default¥AppData¥Local¥NEC¥Rapid¥TSA¥settings¥tsa_predict.conf 」を「C:¥conf¥tsa_predict.conf」へコピーしたものとします。

予測コマンドの実施例は以下となります。

```
rapid /S TSA/exec_predict.rpd -i C:¥data¥predict¥predict_data_cls_prep.csv -m C:¥model¥train_cls -param C:¥conf¥tsa_predict.conf
```

上記コマンドを実行すると、ディレクトリ「C:¥data¥predict」に、簡易予測結果ファイル「predict_data_1cls_prep.result」と詳細予測結果ファイル「predict_data_1cls_prep.dresult」が出力されます。

第5章 CLI 詳細

5.1 CLI 一覧

5.1.1 統計情報取得コマンド（exec_calc_stats）

統計情報取得コマンドの詳細は『表 5-1』となります。

表 5-1 統計情報取得コマンドの詳細

項目	説明		
概要	時系列数値データファイルの統計情報（平均、標準偏差、最小値、最大値）を計算して出力します。ラベルファイルを指定することにより、時系列数値データファイルの指定区間における統計情報を計算することができます。統計情報取得コマンドで取得した情報は前処理のパラメータの指定に使用します。		
構文	rapid ␣/S ␣TSA/exec_calc_stats.rpd ␣<パラメータ>		
パラメータ	-i <時系列数値データファイルパス/ディレクトリパス>	統計情報の計算対象とする、時系列数値データファイルまたは時系列数値データファイルを格納したディレクトリの絶対パスを指定してください。半角スペース区切りで複数指定することで複数のファイルまたはディレクトリを同時に指定できます。ディレクトリを指定した場合は、ディレクトリ内全ての時系列数値データファイルを計算対象とします。	選択必須 （-l とどちらか 1 つ指定） 複数指定可能
	-o <統計情報ファイルパス>	結果を出力する統計情報ファイルの絶対パスを指定してください。指定した場合は、標準出力と統計情報ファイルに結果を出力します。指定しない場合は、標準出力のみに結果を出力します。指定したフォルダが無い場合は自動的に作成されます。	任意
	-l <ラベルファイルパス/ディレクトリパス>	統計情報の計算対象とする、ラベルファイルの絶対パスまたはラベルファイルを格納したディレクトリの絶対パスを指定してください。半角スペース区切りで複数指定することで複数のファイルまたはディレクトリを同時に指定できます。ディレクトリを指定した場合は、ディレクトリ内全てのラベルファイルを計算対象とします。	選択必須 （-i とどちらか 1 つ指定） 複数指定可能
出力	統計情報ファイル	(-o が指定された場合のみ) 時系列数値データから列ごとに算出した統計情報を JSON フォーマットで出力します。詳細は『表 5-2』を参照してください。	
	標準出力	統計情報取得の進捗状況を表示します。詳細は『表 5-2』を参照してください。	
実行例	(Linux の場合) rapid /S TSA/exec_calc_stats.rpd -i /home/user/sample/data.csv -o /home/user/sample/stats.json (Windows の場合) rapid /S TSA/exec_calc_stats.rpd -i C:/sample/data.csv -o C:/sample/stats.json		
備考	処理を中断したいときは Ctrl+C で停止してください。 -l を指定して実行する場合は、時系列数値データファイルとラベルファイルは、必ず同じディレクトリに保存してコマンドを実行してください。		

表 5-2 統計情報取得の進捗状況

進捗出力形式	<Elapsed time>,<File>,<Status>,<Progress in file>
項目	説明
<Elapsed time>	統計情報取得の経過時間です。
<File>	ファイル数を単位とした統計情報取得処理の処理状況です。<処理済・処理中のファイル数>/<対象の全ファイル数>で表します。
<Status>	統計情報取得に対するステータスです。LOAD, SET DATA, CALC STATISTICS があります。
<Progress in file>	統計情報取得の各ファイルに対する処理状況です。
出力例	[Initialize the process.] [Statistics calculating start.] [Elapsed time],[File],[Status],[Progress in file] 0:00:01,1/3,LOAD,0% 0:00:02,1/3,LOAD,100% 0:00:03,1/3,SET DATA,0% 0:00:04,1/3,SET DATA,100% … 0:00:09,3/3,LOAD,0% 0:00:10,3/3,LOAD,100% 0:00:11,3/3,SET DATA,0% 0:00:12,3/3,SET DATA,100% 0:00:12,3/3,CALC STATISTICS,0% 0:00:15,3/3,CALC STATISTICS,25% 0:00:18,3/3,CALC STATISTICS,50% 0:00:21,3/3,CALC STATISTICS,75% 0:00:24,3/3,CALC STATISTICS,100% [Statistics] { "channels":[{ "target":1, "std":<1 列目の標準偏差値>, "mean":<1 列目の平均値>, "src_range":[<1 列目の最小値>, <1 列目の最大値>] }, { "target":2, "std":<2 列目の標準偏差値>, "mean":<2 列目の平均値>, "src_range":[<2 列目の最小値>, <2 列目の最大値>

	]
	},
	…
	],
	"all":{
	"total":<対象サンプル数総計>,
	"std":<全体の標準偏差値>,
	"mean":<全体の平均値>,
	"src_range":[
	<全体の最小値>,
	<全体の最大値>
	]
	}
	}
	[Statistics calculating end.]
	[Finish the process normally.]

⚠注意　標準出力の文字コードについて

・標準出力の文字コードは、Windows の場合は Shift_JIS、Linux の場合は UTF-8 となります。

5.1.2 前処理コマンド（exec_preprocess）

前処理コマンドの詳細は『表 5-3』となります。

表 5-3 前処理コマンドの詳細

項目	説明		
概要	時系列数値データの前処理を行います。		
構文	rapid␣/S␣TSA/exec_preprocess.rpd␣<パラメータ>		
パラメータ	-pre <前処理定義ファイルパス>	前処理定義ファイルの絶対パスを指定してください。	必須
	-i <時系列数値データファイルパス/ディレクトリパス>	前処理の対象とする、時系列数値データファイルまたは時系列数値データファイルを格納したディレクトリの絶対パスを指定してください。半角スペース区切りで複数指定することで複数のファイルまたはディレクトリを同時に指定できます。ディレクトリを指定した場合は、ディレクトリ内全ての時系列数値データファイルを処理対象とします。	選択必須 （-l とどちらか 1 つ指定） 複数指定可能
	-suf <文字列（サフィックス）>	前処理後の時系列数値データファイル名またはラベルファイル名の末尾につける文字列を指定してください。	任意
	-l <ラベルファイルパス/ディレクトリパス>	前処理の対象とする、ラベルファイルまたはラベルファイルを格納したディレクトリの絶対パスを指定してください。半角スペース区切りで複数指定することで複数のファイルまたはディレクトリを同時に指定できます。ディレクトリを指定した場合は、ディレクトリ内全てのラベル	選択必須 （-i とどちらか 1 つ指定） 複数指定可能

		ファイルを対象とします。対象とするラベルファイルによってラベル付けされている時系列数値データファイルに対し前処理が行われます。	
	-o <出力先ディレクトリパス>	前処理済みの時系列数値データファイルとラベルファイルを出力するディレクトリの絶対パスを指定してください。指定しない場合は、-i または-l で指定したディレクトリ、または指定したファイルと同じディレクトリに出力します。	任意
出力	前処理済み時系列数値データファイル	対象の時系列数値データファイルに対して、前処理定義ファイルに記載の前処理を実施した時系列数値データファイルが保存されます。 -o オプションを指定した場合は、指定の出力先ディレクトリに保存されます。-o オプションを指定しない場合は、前処理を行う前の時系列数値データファイルと同じディレクトリに保存されます。	
	ラベルファイル	(-l が指定された場合のみ) 前処理済み時系列数値データファイルに対する新しいラベルファイルです。 -o オプションを指定した場合は、指定の出力先ディレクトリに保存されます。-o オプションを指定しない場合は、前処理を行う前のラベルファイルと同じディレクトリに保存されます。	
	標準出力	前処理の進捗状況を表示します。詳細は『表 5-4』を参照してください。	
実行例	(Linux の場合) rapid /S TSA/exec_preprocess.rpd -pre /home/user/method/method_file.json -i /home/user/data/preprocess/train.csv -suf prep (Windows の場合) rapid /S TSA/exec_preprocess.rpd -pre C:/method/method_file.json -i C:/data/preprocess/train.csv -suf prep		
備考	処理を中断したいときは Ctrl+C で停止してください。 前処理コマンドで出力されるファイル名の規則は『表 5-5』を参照してください。 -l を指定して実行する場合は、時系列数値データファイルとラベルファイルは、必ず同じディレクトリに保存してコマンドを実行してください。		

表 5-4 前処理の進捗状況

進捗出力形式	<Elapsed time>,<File>,<Status>,<Progress in file>
項目	説明
<Elapsed time>	前処理の経過時間です。
<File>	ファイル数を単位とした前処理の処理状況です。<処理済・処理中のファイル数>/<前処理対象の全ファイル数>で表します。
<Status>	前処理中ファイルに対するステータスです。LOAD, PREPROCESS, OUTPUT があります。PREPROCESS では、前処理名も表示します。
<Progress in file>	前処理中ファイルに対する処理状況です。
出力例	[Initialize the process.] [Preprocessing start.] [Elapsed time],[File],[Status],[Progress in file] 0:00:01,1/3,LOAD,0% 0:00:02,1/3,LOAD,100%

	0:00:03,1/3,PREPROCESS(normalize),0% 0:00:08,1/3,PREPROCESS(normalize),40% 0:00:13,1/3,PREPROCESS(normalize),80% 0:00:14,1/3,PREPROCESS(normalize),100% 0:00:15,1/3,PREPROCESS(sampling),0% 0:00:20,1/3,PREPROCESS(sampling),40% 0:00:25,1/3,PREPROCESS(sampling),80% 0:00:26,1/3,PREPROCESS(sampling),100% 0:00:27,1/3,OUTPUT,0% 0:00:30,1/3,OUTPUT,100% 0:00:31,2/3,LOAD,0% 0:00:32,2/3,LOAD,100% 0:00:33,2/3,PREPROCESS(normalize),0% … 0:00:90,3/3,OUTPUT,100% [Preprocessing end.] [Finish the process normally.]

表 5-5 前処理コマンドの出力ファイル名

オプション（-i または -l）	ファイル種別	-suf オプションの指定	出力ファイル名
-i	時系列数値データファイル	なし	<前処理実施前の時系列数値データファイル名>_preprocessed.csv
		あり	<前処理実施前の時系列数値データファイル名>_<-suf で指定した文字列>.csv
	ラベルファイル	なし	出力なし
		あり	出力なし
-l	時系列数値データファイル	なし	<前処理実施前の時系列数値データファイル名>_preprocessed.csv
		あり	<前処理実施前の時系列数値データファイル名>_<-suf で指定した文字列>.csv
	ラベルファイル	なし	<前処理実施前のラベルファイル名>_preprocessed.lab
		あり	<前処理実施前のラベルファイル名>_<-suf で指定した文字列>.lab

5.1.3 学習コマンド（exec_train）

学習コマンドの詳細は『表 5-6』となります。

表 5-6 学習コマンドの詳細

項目	説明		
概要	学習を行います。		
構文	rapid␣/S␣TSA/exec_train.rpd␣<パラメータ>		
パラメータ	-t <学習アルゴリズム>	学習アルゴリズムです。学習に使用するアルゴリズムを 1DCNN、1DDCN、1DOCN から 1 つ指定してください。 指定可能なアルゴリズムの詳細は『表 5-8』を参照してください。	必須
	-mode <学習モード>	学習モードです。分類の場合は cls、回帰の場合は reg を指定してください。学習アルゴリズムと指定可能な学習モードの組み合わせの詳細は『表 5-8』を参照してください。	必須
	-l <ラベルファイルパス/ディレクトリパス>	学習に使用する、ラベルファイルまたはラベルファイルを格納したディレクトリの絶対パスを指定してください。半角スペース区切りで複数指定することで複数のファイルまたはディレクトリを同時に指定できます。ディレクトリを指定した場合は、ディレクトリ内全てのラベルファイルを処理対象とします。	必須
	-m <予測モデルディレクトリパス>	予測モデルの保存先ディレクトリの絶対パスを指定してください。	必須
	-param <学習コンフィグファイルパス>	使用する学習コンフィグファイルの絶対パスを指定してください。指定しない場合はデフォルトのディレクトリにある学習コンフィグファイルを使用します。	任意
	-pre <前処理定義ファイルパス>	学習対象のファイルに前処理を行ってから学習をする場合、前処理定義ファイルの絶対パスを指定してください。	任意
出力	予測モデルファイル	予測モデルの情報を記載したファイルが、-m で指定したディレクトリに保存されます。 ファイル名は model.net です。 -m で指定したディレクトリに同名のファイルが存在した場合は上書きされます。	
	学習情報ファイル	学習時に使用したパラメータを記載したファイルが、-m で指定したディレクトリに保存されます。 ファイル名は model.net.info です。 -m で指定したディレクトリに同名のファイルが存在した場合は上書きされます。	
	学習時コンフィグファイル	学習時に使用した学習コンフィグファイルが、-m で指定したディレクトリに保存されます。 -m で指定したディレクトリに同名のファイルが存在した場合は上書きされます。	
	学習時前処理定義ファイル	-pre オプションを指定して前処理を行った場合、使用した前処理定義ファイルが、-m で指定したディレクトリに保存されます。 -m で指定したディレクトリに同名のファイルが存在した場合は上書きされます。	

	標準出力	学習の進捗状況を表示します。詳細は『表 5-7』を参照してください。
実行例	(Linux の場合) rapid /S TSA/exec_train.rpd -t 1DCNN -mode cls -l /home/user/data/train/train.lab -m /home/user/model/train_01/ -param /home/user/conf/tsa_train.conf (Windows の場合) rapid /S TSA/exec_train.rpd -t 1DCNN -mode cls -l C:/data/train/train.lab -m C:/model/train_01/ -param C:/conf/tsa_train.conf	
備考	処理を中断したいときは Ctrl+C で停止してください。 -l を指定して実行する場合は、時系列数値データファイルとラベルファイルは、必ず同じディレクトリに保存してコマンドを実行してください。	

表 5-7 学習の進捗状況

進捗出力形式	<Elapsed time>,<Total count>,<Learning rate>,<Average error>,<Epoch>,<Status>,<Progress in epoch>
項目	説明
<Elapsed time>	学習の経過時間です。前処理を行う場合、前処理実行の経過時間も含まれます。
<Total count>	ウィンドウ単位での学習処理の累計実行回数です。
<Learning rate>	学習率です。
<Average error>	学習データに対する予測モデルの平均予測誤差を表します。
<Epoch>	学習の反復回数です。<現在の学習反復回数>/<全反復回数>で表します。
<Status>	学習に対するステータスです。LOAD, SET DATA, TRAIN, OUTPUT があります。
<Progress in epoch>	学習の各反復に対する処理状況です。
出力例	※-pre オプションを指定して前処理を行う場合、前処理実行中は前処理コマンド実行時と同じフォーマットの進捗状況が標準出力に出力されます。 [Initialize the process.] [Training start.] [Elapsed time],[File],[Status],[Progress in file] 0:00:01,1/3,LOAD,0% 0:00:02,1/3,LOAD,100% 0:00:03,1/3,PREPROCESS(normalize),0% … 0:01:20,3/3,PREPROCESS(sampling),100% 0:01:21,3/3,SET DATA,0% 0:01:25,3/3,SET DATA,100% [Elapsed time],[Total count],[Learning rate],[Average error],[Epoch],[Status],[Progress in epoch] 0:01:26,0,0.1,0.9,1/100,TRAIN,0% 0:01:30,800,0.1,0.9,1/100,TRAIN,80% 0:01:31,1000,0.1,0.9,1/100,TRAIN,100% 0:01:32,1000,0.099,0.89,2/100,TRAIN,0% 0:01:35,1300,0.099,0.89,2/100,TRAIN,30% … 0:10:00,99600,0.001,0.01,100/100,TRAIN,60%

	0:10:02,100000,0.001,0.01,100/100,TRAIN,100% 0:10:03,100000,0.001,0.01,100/100,OUTPUT,0% 0:10:05,100000,0.001,0.01,100/100,OUTPUT,100% [Training end.] [Finish the process normally.]

⚠注意　バッチサイズを 2 以上にした場合の<Total count>の値について

V2.2 では、バッチサイズの件数のデータを一度に学習するミニバッチ学習をサポートしているため、学習コンフィグファイルのバッチサイズに 2 以上の値を指定した場合、進捗状況の Total count の値は、V1.2 より少ない値が表示されます。

(例)
・バッチサイズ 1 の場合
X:XX:XX,100000,0.001,0.01,100/100,TRAIN,100%

・バッチサイズ 8 の場合
X:XX:XX,12500,0.001,0.01,100/100,TRAIN,100%

⚠注意　<Average error>の値について

V2.2 では、V1.2 と異なる誤差の計算方法を使用しているため、同じ学習を行った場合でも、進捗状況の Average error の値が変わります。

学習アルゴリズムについて

-t に指定可能な学習アルゴリズムと-mode で指定可能な学習モードは以下の『表 5-8』の通りです。

表 5-8 指定可能な学習アルゴリズム

学習アルゴリズム	学習モード		説明	選択基準
	分類	回帰		
1DCNN	○	○	畳み込みニューラルネットワークの手法を利用する学習アルゴリズムです。	学習時間の短縮を優先したい場合に使用してください。
1DDCN	○	○	1DCNN よりもさらに深いニューラルネットワークを構築できる学習アルゴリズムです。 本アルゴリズムは V2.2 より使用可能です。	精度の向上を優先したい場合に使用してください。
1DOCN	○	×	正常データのみを学習させることで、正常と異常を分類できる予測モデルを作成する学習アルゴリズムです。 本アルゴリズムは V2.2 より使用可能です。	十分な数の異常データがなく、1DCNN や 1DDCN による正常／異常分類の予測モデル作成が困難な場合に使用してください。

5.1.4 予測コマンド（exec_predict）

予測コマンドの詳細は『表 5-9』となります。

表 5-9 予測コマンドの詳細

項目	説明		
概要	予測を行います。		
構文	rapid␣/S␣TSA/exec_predict.rpd␣<パラメータ>		
パラメータ	-l <ラベルファイルパス/ディレクトリパス>	予測対象の時系列数値データファイルに対する、ラベルファイルまたはラベルファイルを格納したディレクトリの絶対パスを指定してください。半角スペース区切りで複数指定することで複数のファイルまたはディレクトリを同時に指定できます。ディレクトリを指定した場合は、ディレクトリ内全てのラベルファイルを処理対象とします。	選択必須 （-i とどちらか 1 つ指定） 複数指定可能
	-i <時系列数値データファイルパス/ディレクトリパス>	予測対象の、時系列数値データファイルまたは時系列数値データファイルを格納したディレクトリの絶対パスを指定してください。半角スペース区切りで複数指定することで複数のファイルまたはディレクトリを同時に指定できます。ディレクトリを指定した場合は、ディレクトリ内全ての時系列数値データファイル(csv ファイル)を処理対象とします。	選択必須 （-l とどちらか 1 つ指定） 複数指定可能
	-m <予測モデルディレクトリパス>	使用する予測モデルが保存されたディレクトリの絶対パスを指定してください。	必須
	-suf <文字列（サフィックス）>	結果ファイル名の末尾につける文字列を指定してください。	任意
	-o <出力先ディレクトリパス>	結果ファイルを出力するディレクトリの絶対パスを指定してください。指定しない場合は、-i または-l で指定したディレクトリ、または指定したファイルと同じディレクトリに出力します。	任意
	-param <予測コンフィグファイルパス>	使用する予測コンフィグファイルの絶対パスを指定してください。指定しない場合はデフォルトのディレクトリにある予測コンフィグファイルを使用します。	任意
	-pre <前処理定義ファイルパス>	予測対象のファイルに前処理を行ってから予測をする場合、前処理定義ファイルの絶対パスを指定してください。	任意
出力	簡易予測結果ファイル	時系列数値データファイルに対して予測モデルで予測した結果を出力したファイルです。 -o オプションを指定した場合は、指定の出力先ディレクトリに保存されます。-o オプションを指定しない場合は、予測対象の時系列数値データファイルまたはラベルファイルと同じディレクトリに出力されます。 複数の時系列数値データファイルに対して予測した場合、それぞれの時系列数値データファイルに対応する予測結果ファイルが出力されます。 フォーマットは『表 6-3』を参照してください。	
	詳細分類結果ファイ	学習モードに分類を指定して学習した予測モデルを使って予測した場合	

	ル	のみに出力される結果ファイルです。 -o オプションを指定した場合は、指定の出力先ディレクトリに保存されます。-o オプションを指定しない場合は、予測対象の時系列数値データファイルまたはラベルファイルと同じディレクトリに出力されます。 複数の時系列数値データファイルに対して予測した場合、それぞれの時系列数値データファイルに対応する予測結果ファイルが出力されます。 フォーマットは『表 6-4』を参照してください。
	標準出力	予測の進捗状況を表示します。詳細は『表 5-10』を参照してください。
実行例	(Linux の場合) rapid /S TSA/exec_predict.rpd -i /home/user/data/predict/predict.csv -m /home/user/model/train_01/ -param /home/user/conf/tsa_predict.conf (Windows の場合) rapid /S TSA/exec_predict.rpd -i C:/data/predict/predict.csv -m C:/model/train_01/ -param C:/conf/tsa_predict.conf	
備考	処理を中断したいときは Ctrl+C で停止してください。 時系列数値データファイルとラベルファイルは、必ず同じディレクトリに保存してコマンドを実行してください。	

表 5-10 予測の進捗状況

進捗出力形式	<Elapsed time>,<Total count>,<FIle>,<Status>,<Progress in file>
項目	説明
<Elapsed time>	予測の経過時間です。前処理を行う場合、前処理実行の経過時間も含まれます。
<Total count>	ウィンドウ単位での予測処理の累計実行回数です。
<File>	ファイル数を単位とした前処理の処理状況です。<処理済・処理中のファイル数>/<予測対象の全ファイル数>で表します。
<Status>	予測に対するステータスです。LOAD, SET DATA, PREDICTION があります。
<Progress in file>	予測中ファイルに対する処理状況です。
出力例	[Initialize the process.] [Predicting start.] [Elapsed time],[Total count][File],[Status],[Progress in file] 0:00:01,0,1/3,LOAD,0% 0:00:02,0,1/3,LOAD,100% 0:00:03,0,1/3,PREPROCESS(normalize),0% … 0:00:20,0,1/3,PREPROCESS(sampling),100% 0:00:21,0,1/3,SET DATA,0% 0:00:22,0,1/3,SET DATA,100% 0:00:23,0,1/3,PREDICTION,0% 0:00:25,400,1/3,PREDICTION,40% 0:00:28,1000,1/3,PREDICTION,100% 0:00:29,1000,2/3,LOAD,0% 0:00:30,1000,2/3,LOAD,100%

	0:00:31,1000,2/3,PREPROCESS(normalize),0% … 0:01:24,3000,3/3,PREDICTION,100% [Predicting end.] [Finish the process normally.]

予測コマンドで出力されるファイル名の規則は『表 5-11』の通りです。

表 5-11 予測コマンドの出力ファイル名

オプション（-i または -l）	ファイル種別	-suf オプションの指定	出力ファイル名
-i	簡易予測結果ファイル	なし	<時系列数値データファイル名>.result
		あり	<時系列数値データファイル名>_<-suf で指定した文字列>.result
	詳細分類結果ファイル	なし	<時系列数値データファイル名>.dresult
		あり	<時系列数値データファイル名>_<-suf で指定した文字列>.dresult
-l	簡易予測結果ファイル	なし	<ラベルファイル名>.result
		あり	<ラベルファイル名>_<-suf で指定した文字列>.result
	詳細分類結果ファイル	なし	<ラベルファイル名>.dresult
		あり	<ラベルファイル名>_<-suf で指定したファイル名>.dresult

5.1.5 オンライン予測

オンライン予測機能の概要を『図 5-1』に記載します。
本機能は TSA CLI の予測コマンドを常駐プロセス化することにより、プロセスの初期化に必要な時間を短縮し、高速に予測結果を返却することができる機能です。
この TSA 常駐プロセスは、複数の予測モデルファイルを読み込んでキャッシュし、予測リクエストごとで予測モデルを切り替えることもできます。予測リクエストで指定されたモデルがキャッシュ済みのモデルの場合は、モデルファイル読み込み処理もスキップされるため、その分高速に予測結果を返却することが可能です。なお、予測リクエストで指定されたモデルがキャッシュ済みのモデルでない場合は、指定されたモデルファイルを読み込んでキャッシュに追加した上で、予測処理を行います。

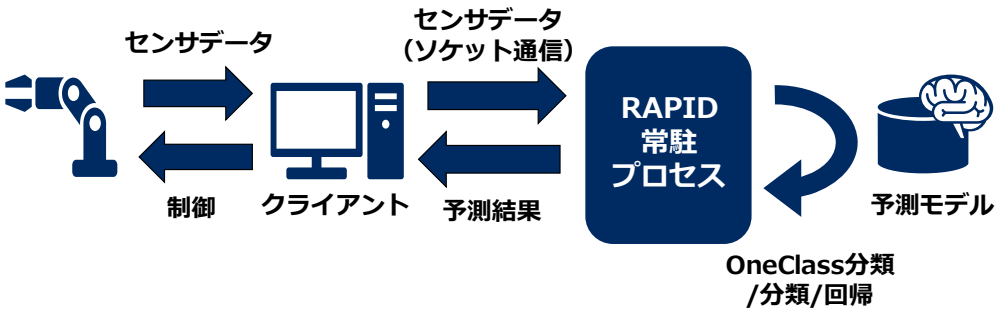

図 5-1 オンライン予測機能の概要図

オンライン予測の使い方

オンライン予測を使用する場合は、TSA 常駐プロセスへ予測リクエストを送信し、レスポンスを受け取るクライアントが必要になります。

表 5-12 クライアント(client.py)の例

```
# -*- coding: utf-8 -*-
import json
import socket

EOL = "¥r¥n"
RESPONSE_SIZE_DIGIT = 9
BUF_SIZE = 4096
ENCODING = "utf-8"

if __name__ == "__main__":
    ip = "127.0.0.1"
    port = 34999

    request = {}
    request["model"] = "train_cls"
    request["data"] = [
            {"id": "0:01:00", "value": [10.0, 0.3]},
            {"id": "0:02:00", "value": [9.0, 0.45]},
            {"id": "0:03:00", "value": [8.0, 0.4]},
            {"id": "0:04:00", "value": [7.0, 0.43]},
            {"id": "0:05:00", "value": [6.0, 0.6]},
            {"id": "0:06:00", "value": [5.0, 0.3]},
            {"id": "0:07:00", "value": [5.0, 0.34]},
            {"id": "0:08:00", "value": [2.0, 0.42]},
            {"id": "0:09:00", "value": [1.0, 0.51]}
        ]
    request["preprocess"] = {
        "unity":[{
                "method": "sampling",
                "param":{"type":"linear","rate":10.0}
            }
        ]
    }

    request_json = json.dumps(request) + EOL
```

```
    response_json = None
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        sock.connect((ip, port))
        sock.send(request_json.encode(ENCODING))
        response_size = int(sock.recv(RESPONSE_SIZE_DIGIT))
        response = bytes()
        while len(response) < response_size:
            response = response + sock.recv(min(BUF_SIZE, response_size - len(response)))
        response_json = response.decode(ENCODING)
    except socket.error as e:
        exit(1)
    except ValueError as e:
        exit(2)
    finally:
        sock.close()

    if response_json:
        print(response_json)
```

上記クライアント(client.py)を使用した TSA 常駐プロセスの使用例は以下の通りです。

1. TSA 常駐プロセス起動コマンドを実行します。

```
rapid ␣ /S ␣ TSA/socket_daemon.rpd -ip 127.0.0.1 -port 34999 -param /home/user/conf/tsa_online_predict.conf
```

2. クライアントを実行し、起動した TSA 常駐プロセスにリクエストを送り、解析結果を受け取ります。

```
python client.py
```

常駐プロセス起動コマンド

TSA 常駐プロセスの起動を行います。コマンドの詳細は『表 5-13』となります。

表 5-13 常駐プロセス起動コマンドの詳細

項目	説明		
概要	オンライン予測を実行する常駐プロセスを起動します。		
構文	rapid␣/S␣TSA/socket_daemon.rpd␣<パラメータ>		
パラメータ	-ip	常駐プロセスとメッセージをやり取りするための TCP ソケットの IP アドレス（IPv6 は未サポート）を指定してください。	必須
	-port	常駐プロセスとメッセージをやり取りするための TCP ソ	必須

		ケットのポート番号です。20000～65535 の範囲で指定してください。	
	-param	使用するオンライン予測コンフィグファイルの絶対パスを指定してください。指定がない場合はデフォルトのディレクトリにあるオンライン予測コンフィグファイルを使用します。	任意
出力	標準出力	『起動』、『準備完了（ソケット接続待ちの開始）』、『終了』の状態を出力します。その際、OS が割り当てたプロセス ID (PID)があわせて出力されます。その他、エラーが発生した場合はそのログを出力します。 例） > rapid /S TSA/socket_daemon.rpd -ip 127.0.0.1 -port 34999 [Socket daemon start (PID = 2749).] [Socket daemon is ready.] [Finish the process normally.]	
実行例	(Linux の場合) rapid␣/S␣TSA/socket_daemon.rpd -ip 127.0.0.1 -port 34999 -param /home/user/conf/tsa_online_predict.conf		
	(Windows の場合) rapid␣/S␣TSA/socket_daemon.rpd -ip 127.0.0.1 -port 34999 -param C:/User/user/conf/tsa_online_predict.conf		

備考	IP アドレス、ポートの異なる常駐プロセスを複数起動することが可能です。
	常駐プロセスの起動は、オンライン予測コンフィグファイルで指定するモデル格納ディレクトリにアクセス可能なユーザで実行してください。
	常駐プロセスを停止する場合は、起動したユーザまたは管理者権限を持つユーザでプロセスを kill するか、確立した TCP ソケットをクライアント側で切断してください。
	例）プロセス ID が 11956 の場合
	
	[Windows]
	`taskkill -F /PID 11956`
	
	[Linux]
	`kill -9 11956`		

リクエストの詳細

TSA 常駐プロセスによる予測リクエストを行います。予測リクエストの詳細は『表 5-14』となります。

表 5-14 予測リクエストの詳細

項目	説明
概要	常駐プロセスへ予測リクエストを送信し、オンライン予測を実行します。予測リクエスト及び予測結果のレスポンスは、JSON 形式のメッセージとして TCP ソケット経由で送受信されます。
構文	{

<table>
<tr><td></td><td colspan="3">"model": "<モデル名>",
"preprocess": {
"unity": [
{<前処理定義 1>},
{<前処理定義 2>},
...
{<前処理定義 N>}
]
},
"data": [
{ "id": <ID 1>, "value": [<value 1-1>, <value 1-2>, ...] },
{ "id": <ID 2>, "value": [<value 2-1>, <value 2-2>, ...] },
...
{ "id": <ID M>, "value": [<value M-1>, <value M-2>, ...] }
],
"alg_param": {
"slide_size": 1,
"1docn_tolerance": 1.0,
"1docn_filter_size": 1
}
}¥n

※ここでは見やすさのために改行していますが、途中で改行は含めず末尾にのみ改行(LF または CRLF)を含めて一行で送信してください。
※alg_param のうち、1docn_tolerance と 1docn_filter_size は、1DOCN の予測モデルを用いる場合のみ指定が必須となります。</td></tr>
<tr><td rowspan="4">パラメータ</td><td>model</td><td>予測に用いるモデルファイルの保存されたディレクトリ名を指定してください。オンライン予測コンフィグファイルのパラメータ[MODEL_DIR]直下に格納されたディレクトリ名が指定可能です。相対パスでの指定はできません。</td><td>必須</td></tr>
<tr><td>preprocess</td><td>前処理定義を指定します。前処理定義ファイルのフォーマットは『表 6-5』を参照してください。</td><td>任意</td></tr>
<tr><td>data</td><td>予測対象の時系列数値データです。1 行分のデータごとに id と value の配列で指定してください。id は必ず昇順で指定する必要があります。id と value の形式については『表 6-1』を参照してください。</td><td>必須</td></tr>
<tr><td>alg_param</td><td>予測に関するパラメータを指定します。指定可能なパラメータは『表 6-15』を参照してください。指定しなかった場合はデフォルトの値を使用します。</td><td>任意</td></tr>
<tr><td>出力</td><td colspan="3">予測結果を TCP ソケット経由でクライアントに返却します。
出力文字列の先頭 9 バイトは以降の出力のバイト数となり(例:000123456)、クライアントはこの値により読み込むべきバイト数を判断します。
それ以降は、実行ステータス(TSA_status)と予測結果を JSON メッセージに変換した文字列になります。
また、前処理の経過、予測の経過はログに出力されます。</td></tr>
<tr><td>実行例</td><td colspan="3"></td></tr>
</table>

備考	文字コードは UTF-8 になります。

リクエストの実行例と出力例

0～255 のデータを-1.0～1.0 に正規化し、モデル「model_1dcnn_cls」で予測（分類）を行う場合

```
{
    "model": "model_1dcnn_cls",
    "preprocess": {
        "unity": [
            {
                "method": "normalize",
                "param": [ { "target": 0, "src_range": [0, 255], "dst_range": [-1.0, 1.0] } ]
            },
        ]
    },
    "data": [
        {"id": "0.000000", "value": [0.84147098480790, 0.90929742682568]},
        {"id": "0.000001", "value": [0.90929742682568, -0.75680249530793]},
        ...,
        {"id": "0.001000", "value": [0.14987720966295, 0.29636857870939]}
    ],
    “alg_param”: {
        “slide_size”: 8
    }
}
```

予測リクエスト（分類）の実行例

```
000169813
{
    "TSA_status":<ステータス>,
    "result_num":<予測結果（ウィンドウ）の総数>,
    "result_list": [
        {
            "window": { "start_time": <始点 ID >, "end_time"   : <終点 ID > },
            "result": "<分類結果>",
            “confidence”:{
                “ラベル 1”:確信度 1,
                “ラベル 2”:確信度 2,
                ...,
                “ラベル M”:確信度 M
```

```
            }
        },
        ...,
        {
            "window": { "start_time": <始点 ID >, "end_time"   : <終点 ID > },
            "result": "<分類結果>",
            “confidence”:{
                “ラベル 1”:確信度 1,
                “ラベル 2”:確信度 2,
                ...,
                “ラベル M”:確信度 M
            }
        }
    ]
}
```

予測結果レスポンス（分類）の出力フォーマット例

⚠注意　リクエストの改行・スペースについて

リクエストの実行例と出力例では改行・スペースを含んでいますが、実際のリクエスト、レスポンスに改行・スペースは含まれません。

データのサンプリング補正を 0.1 秒周期で行ってから、モデル「model_1dcnn_reg」で予測を行う場合（回帰）

```
{
    "model": "model_1dcnn_reg",
    "preprocess": {
        "unity": [
            {
                "method": "sampling",
                "param": { “type”:"linear", “rate”:0.1 }
            },
        ]
    },
    "data": [
        {"id": "0.000000", "value": [0.84147098480790, 0.90929742682568]},
        {"id": "0.000001", "value": [0.90929742682568, -0.75680249530793]},
        ...,
        {"id": "0.001000", "value": [0.14987720966295, 0.29636857870939]}
    ],
    “alg_param”: {
```

```
            “slide_size”: 8
        }
}
```

予測リクエスト（回帰）の実行例

```
000169613
{
        "TSA_status":<ステータス>,
        "result_num":<予測結果（ウィンドウ）の総数>,
        "result_list": [
                {
                        "window": { "start_time": <始点 ID >, "end_time"    : <終点 ID > },
                        "result": "<出力値>"
                },
                ...,
                {
                        "window": { "start_time": <始点 ID >, "end_time"    : <終点 ID > },
                        "result": "<出力値>"
                }
        ]
}
```

予測結果レスポンス（回帰）の出力のフォーマット

0～255 のデータを-1.0～1.0 に正規化し、モデル「model_1docn_cls」で予測を行う場合（OneClass 分類）

```
{
        "model": "model_1docn_cls",
        "preprocess": {
                "unity": [
                        {
                                "method": "normalize",
                                "param": [ { "target": 0, "src_range": [0, 255], "dst_range": [-1.0,
1.0] } ]
                        },
                ]
        },
        "data": [
                {"id": "0.000000", "value": [0.84147098480790, 0.90929742682568]},
                {"id": "0.000001", "value": [0.90929742682568, -0.75680249530793]},
                ...,
```

```
            {"id": "0.001000", "value": [0.14987720966295, 0.29636857870939]}
      ],
      “alg_param”: {
            “slide_size”: 8,
            "1docn_tolerance": 1.0,
            "1docn_filter_size": 1
      }
}
```

予測リクエスト（OneClass 分類）の実行例

```
000169713
{
      "TSA_status":<ステータス>,
      "result_num":<予測結果（ウィンドウ）の総数>,
      "result_list": [
            {
                  "window": { "start_time": <始点 ID >, "end_time"    : <終点 ID > },
                  "result": "<出力値>",
                   “anomaly”:”<異常度>”
            },
            ...,
            {
                  "window": { "start_time": <始点 ID >, "end_time"    : <終点 ID > },
                  "result": "<分類結果>",
                  “anomaly”:”<異常度>”
            }
      ]
}
```

予測結果レスポンス（OneClass 分類）の出力のフォーマット

5.2 使用可能文字種別

使用可能な文字種別の一覧は『表 5-15』の通りです。

表 5-15 使用可能文字種別一覧

		半角英数字	マルチバイト文字	スペース	記号				
					¥	:	/	.	*?"<>|,
時系列数値データファイル	ID name	○	○	○	×	×	×	×	×
	value name	○	○	○	×	×	×	×	×
ラベルファイル	data path	○	○	○	×	×	×	○	×
	ラベル	○	×	×	×	×	×	×	×
コマンドオプション	サフィックス(-suf)	○	○	○	×	×	×	×	×
	ファイル絶対パス	○	○	○	Linux × Windows ○	Linux × Windows ○	○	○	×
コマンドオプション、コンフィグファイル	ディレクトリ絶対パス	○	○	○	Linux × Windows ○	Linux × Windows ○	○	○	×

⚠注意

・ID name、value name、ラベルに指定できる文字列長は実行コンフィグファイルの設定値に依存します。
・data path、サフィックス、ファイルおよびディレクトリの絶対パスに指定できる文字列長はOS の規定に依存します。
・ID name、value name、data path、ラベルは、ダブルクォーテーション(" ")で囲ってください。
・ファイルおよびディレクトリの絶対パスは、スラッシュ(/)区切りにしてください。Windowsの場合は、バックスラッシュ(¥)を使うこともできます。
・ディレクトリの絶対パスの末尾には、必ずスラッシュ(/)をつける必要はありません。
・スペースは、半角、全角スペースです。

5.3 Exit コード一覧

Exit コード一覧は『表 5-16』の通りです。

表 5-16 Exit コード一覧

Exit コード名	数値	概要
END_NORMAL	0	処理が正常に終了しました。
CONFIG_ERROR	1	コンフィグファイルの記述が不正な場合に発生します。 ログファイルにエラーの詳細が記載されていますので確認してください。
COMMAND_ERROR	2	コマンドに指定されたパラメータが不正な場合に発生します。 ログファイルにエラーの詳細が記載されていますので確認してください。
PREPROCESS_ERROR	3	前処理定義ファイルの記述が不正な場合に発生します。 ログファイルにエラーの詳細が記載されていますので確認してください。
DATA_ERROR	4	時系列数値データファイルの記述が不正な場合に発生します。 ログファイルにエラーの詳細が記載されていますので確認してください。
LABEL_ERROR	5	ラベルファイルの記述が不正な場合に発生します。 ログファイルにエラーの詳細が記載されていますので確認してください。
MODEL_ERROR	6	予測モデルのファイルの記述が不正な場合に発生します。 ログファイルにエラーの詳細が記載されていますので確認してください。
LICENSE_ERROR	7	GPU を利用可能なコマンドにおいて、GPU ライセンスが無効の状態で動作モードに GPU を指定した場合に発生します。
NOGPU_FOUND	8	GPU が利用可能なコマンドにおいて、使用可能な GPU が見つからない場合に発生します。
TRAIN_ERROR	9	学習時に問題が発生し学習に失敗した時に発生します。
SOCKET_ERROR	10	オンライン予測の TCP ソケット通信でエラーとなった場合に発生します。
REQUEST_ERROR	11	オンライン予測の予測リクエストメッセージに問題があった場合に発生します。
SYSTEM_ERROR	127	不明なエラーです。 詳細はログファイルを参照してください。

第6章 ファイル

6.1 時系列数値データファイル

時系列数値データファイルのフォーマットの詳細は『図 6-1』の通りです。

```
[Header]"<ID name>","<value name 1>","<value name 2>",…,"<value name M>"¥n
<ID 1>,<value 11>,<value 12>,…,<value 1M>¥n
<ID 2>,<value 21>,<value 22>,…,<value 2M>¥n
….
```

図 6-1 時系列数値データファイルのフォーマット

時系列数値データファイルのフォーマットの詳細は『表 6-1』の通りです。

表 6-1 フォーマット説明

項目	説明	
概要	時系列数値データを記載したファイルです。	
ファイル形式	CSV 形式	
拡張子	csv	
文字コード	UTF-8（BOM なし）	
改行コード	LF	
要素説明	<ID name>	時系列数値データの各行の識別子の名称を指定してください。 指定した文字列はダブルクォーテーション（" "）で囲ってください。
	<value name>	時系列数値データの各列の名称を指定してください。 指定した文字列はダブルクォーテーション（" "）で囲ってください。
	<ID>	1 行分のデータの識別子です。 秒または時刻（hh:mm:ss.ffffff）の形式の指定が可能です。同一の時系列数値データファイル内では形式を揃えて指定してください。 <ID>は必ず昇順となるよう指定してください。 指定できる値の範囲は以下の通りです。 ・秒：[0, 2147483647.999999]の実数。 小数点以下は 6 桁まで指定可能。 ・時刻：[00:00:00.000000, 99:59:59.999999]
	<value>	各<ID>における各<value name>の値です。 実数値の指定が可能です。値が存在しない場合は指定しなくても構いません。
補足	各項目に指定できる文字種別は『表 5-15』を参照してください。 ID の指定範囲を超えるデータを記載したい場合は、別のファイルに分けて記載してください。 ファイルを分けた場合は、ID を指定可能範囲内に変換して記載してください。	

	<value>の一部に値を指定しない場合、前処理コマンドまたは学習・予測コマンドにおける前処理オプションを用いて、サンプリング補正を実行してから、学習・予測に使用してください。

⚠注意 時系列数値データファイルの列の並びについて

時系列数値データファイルは、学習と予測に使う全てのファイル間で、列の並び順をそろえてください。
具体的には、以下の全てで、同じ列の名称・並び順になっている必要があります。
・学習用データファイル同士
・予測用データファイル同士
・学習用データファイルと予測用データファイル

6.2 ラベルファイル

時系列数値データファイルに対するラベルを記載したファイルです。

時系列数値データファイルとラベルファイルは必ず同じディレクトリに保存してください。

ラベルファイルのフォーマットは『図 6-2』の通りです。

```
[DataPath]"<data path>"¥n
<始点 ID 1>,<終点 ID 1>,"<ラベル 1>"¥n
<始点 ID 2>,<終点 ID 2>,"<ラベル 2>"¥n
<始点 ID 3>,<終点 ID 3>,¥n
<始点 ID 4>,<終点 ID 4>,"<ラベル 4>"¥n
….
```

図 6-2 ラベルファイルのフォーマット

ラベルファイルのフォーマットの詳細は『表 6-2』の通りです。

表 6-2 ラベルファイルのフォーマット説明

項目	説明	
概要	時系列数値データファイルの各区間とそれに対するラベルを記載したファイルです。	
ファイル形式	CSV 形式	
拡張子	lab	
文字コード	UTF-8（BOM なし）	
改行コード	LF	
要素説明	<data path>	ラベル付け対象とする時系列数値データファイル名を指定してください。 指定した文字列はダブルクォーテーション（" "）で囲ってください。
	<始点 ID>	時系列数値データファイルのラベル付け対象区間の始点となる行の識別子（ID）を指定してください。

		秒または時刻（hh:mm:ss.ffffff）の形式の指定が可能です。 形式はラベル付け対象の時系列数値データファイルと同一のものを指定してください。
	<終点 ID>	時系列数値データファイルのラベル付け対象区間の終点となる行の識別子（ID）を指定してください。 秒または時刻（hh:mm:ss.ffffff）の形式の指定が可能です。 形式はラベル付け対象の時系列数値データファイルと同一のものを指定してください。 <終点 ID>は、同一行の<始点 ID>より大きな値を指定してください。
	<ラベル>	時系列数値データファイルの<始点 ID>から<終点 ID>までの区間に対するラベルを指定してください。 前処理コマンドまたは予測コマンドにラベルファイルを使用する場合は、指定しないことも可能です。 指定した文字列はダブルクォーテーション（" "）で囲ってください。
補足	同一ラベルファイル内では、必ず<始点 ID>が昇順となるよう指定してください。 各項目に指定できる文字種別は『表 5-15』を参照してください。	

時系列数値データとラベルファイルの関係は『図 6-3』の通りです。
ラベルファイルの各行の<始点 ID>から<終点 ID>までの区間に含まれる時系列数値データが、学習・予測に用いられます。

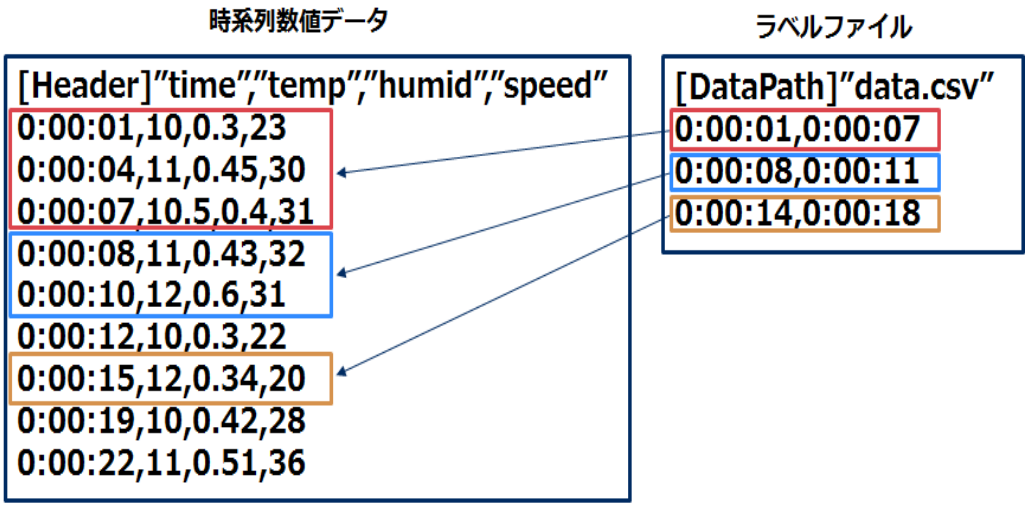

図 6-3 時系列数値データとラベルファイルの関係

⚠注意　始点 ID、終点 ID の形式について

ID を実数（秒形式）とした場合、、時系列数値データファイルとラベルファイルの対応を取る ID として、整数の ID と同値な実数の ID を互いに指定することができます。

(例)
時系列数値データファイルの ID を”1”に対して、ラベルファイルの ID を”1.000000”と指定。

6.3 結果ファイル

6.3.1 簡易予測結果ファイル

簡易予測結果ファイルのフォーマットは『図 6-4』の通りです。

```
[DataPath]"<data path>"¥n
<始点 ID 1>,<終点 ID 1>,"<ラベル 1>"¥n
<始点 ID 2>,<終点 ID 2>,"<ラベル 2>"¥n
<始点 ID 3>,<終点 ID 3>,"<ラベル 3>"¥n
…
```

図 6-4 簡易予測結果ファイルのフォーマット

簡易予測結果ファイルのフォーマットの詳細は『表 6-3』の通りとなります。

表 6-3 簡易予測結果ファイルのフォーマット詳細

項目	説明	
概要	時系列数値データに対する予測結果を記載したファイルです。	
ファイル形式	CSV 形式	
拡張子	result	
文字コード	UTF-8（BOM なし）	
改行コード	LF	
要素説明	<data path>	予測対象の時系列数値データファイル名です。
	<始点 ID>	時系列数値データファイルの予測対象区間の始点の識別子（ID）です。予測対象の時系列数値データファイルと同じ形式の識別子で記載されます。
	<終点 ID>	時系列数値データファイルの予測対象区間の終点の識別子（ID）です。予測予測対象の時系列数値データファイルと同じ形式の識別子で記載されます。
	<ラベル>	時系列数値データファイルの<始点 ID>から<終点 ID>までの区間に対する予測結果ラベルが記載されます。使用した予測モデルの学習時の mode が分類の場合は確信度が最大のラベル、回帰の場合は実数値が記載されます。

補足	

6.3.2 詳細予測結果ファイル

詳細予測結果ファイルのフォーマットは『図 6-5』の通りです。

```
[DataPath]"<data path>"¥n
<始点 ID 1>,<終点 ID 1>,"<ラベル 11>",<確信度 11>,…,"<ラベル 1M>",<確信度 1M>¥n
<始点 ID 2>,<終点 ID 2>,"<ラベル 21>",<確信度 21>,…,"<ラベル 2M>",<確信度 2M>¥n
….
```

図 6-5 詳細予測結果ファイルのフォーマット

詳細予測結果ファイルのフォーマットの詳細は『表 6-4』の通りです。

表 6-4 詳細予測結果ファイルのフォーマット詳細

項目	説明	
概要	時系列数値データに対する予測結果の詳細を記載したファイルです。 学習時の mode が分類の予測モデルを使用して予測を実施したときのみ出力されます。	
ファイル形式	CSV 形式	
拡張子	dresult	
文字コード	UTF-8（BOM なし）	
改行コード	LF	
要素説明	<data path>	予測対象の時系列数値データファイル名です。
	<始点 ID>	時系列数値データファイルの予測対象区間の始点の識別子（ID）です。予測対象の時系列数値データファイルと同じ形式の識別子で記載されます。
	<終点 ID>	時系列数値データファイルの予測対象区間の終点の識別子（ID）です。予測対象の時系列数値データファイルと同じ形式の識別子で記載されます。
	<ラベル>	時系列数値データファイルの<始点 ID>から<終点 ID>までの区間に対する予測結果ラベルが記載されます。各ラベルは確信度が高い順に記載されます。
	<確信度>	予測結果ラベルに対する確信度が記載されます。
補足		

OneClass 分類の場合

OneClass 分類の場合、正常ラベル OK の確信度には予測コンフィグファイルの[1DOCN_TOLERANCE]に設定した値が、異常ラベル NG の確信度には予測モデルで計算した異常度が出力されます。

```
[DataPath]"<data path>"¥n
<始点 ID 1>,<終点 ID 1>,"NG",<異常度 1>,"OK",[1DOCN_TOLERANCE]¥n
<始点 ID 2>,<終点 ID 2>,"NG",<異常度 2>,"OK",[1DOCN_TOLERANCE]¥n
<始点 ID 3>,<終点 ID 3>,"OK",[1DOCN_TOLERANCE],"NG",<異常度 3>¥n
….
```

6.4 前処理定義ファイル

6.4.1 フォーマット

前処理定義ファイルのフォーマットは『図 6-6』の通りです。

```
{
  "unity":
  [
    {<前処理定義 1(method, param)>},
    {<前処理定義 2(method, param)>}
  ]
}
```

図 6-6 前処理定義ファイルのフォーマット

前処理定義ファイルのフォーマットの詳細は『表 6-5』の通りです。

表 6-5 前処理定義ファイルのフォーマット詳細

項目	説明	
概要	時系列数値データファイルに対する前処理方法を記載したファイルです。	
ファイル形式	JSON 形式	
拡張子	json	
文字コード	UTF-8（BOM なし）	
改行コード	LF	
要素説明	前処理定義 （method, param）	時系列数値データファイルに対する前処理名（method）とそのパラメータ（param）を記載してください。記載された順に前処理が実行されます。 指定可能な前処理名（method）は、正規化（normalize）、標準化（standardize）、サンプリング補正（sampling）の 2 種類です。 各前処理のパラメータ（param）については、『表 6-6』、『表 6-7』、『表 6-8』を参照してください。
補足		

6.4.2 処理一覧

正規化

正規化の処理については『表 6-6』の通りです。

表 6-6 正規化処理

項目	説明		
概要	時系列数値データを列ごとに正規化します。		
要素説明	名称	値	説明
	target	0 以上の整数	正規化する対象の列番号を指定してください。0 を指定すると全ての列が対象となります。0 以外の値を target に指定した設定がある場合は、それが優先されます。
	src_range	実数のリスト	列単位での正規化前のデータの最小値と最大値の組を指定してください。
	dst_range	実数のリスト	列単位での正規化後のデータの最小値と最大値の組を指定してください。指定しない場合は最小値が-1、最大値が+1となります。

定義例	例）10 列目以外の値を[0,255]の範囲から[-1,1]の範囲へ正規化。10 列目の値を[-1,1]の範囲から[0,1]の範囲へ正規化。
	{
	"method":"normalize",
	"param":[
	{"target":0, "src_range":[0,255]},
	{"target":10, "src_range":[-1,1], "dst_range":[0,1]}
	]
	}		

備考			

標準化

標準化の処理については『表 6-7』の通りです。

表 6-7 標準化処理

項目	説明		
概要	時系列データを列ごとに標準化します。時系列データの各列の値の平均とばらつきを均一化することで、学習・予測における列間の影響力の差を是正します。		
要素説明	名称	値	説明
	target	0 以上の整数	標準化する対象の列番号を指定してください。0 を指定すると全ての列が対象となります。0 以外の値を target に指定した設定がある場合は、それが優先されます。
	mean	実数	列単位での標準化前のデータの平均値を指定してください。

	std	正の実数	列単位での標準化前のデータの標準偏差を指定してください。

定義例	例）8 列目と 10 列目の値を標準化する場合：
	{
	"method":"standardize",
	"param":[
	{"target":8, "mean":3.6345789, "std":1.2430973},
	{"target":10, "mean":8.0566524, "std":5.8943624}
	]
	}		

備考			

サンプリング補正

サンプリング補正の処理については『表 6-8』の通りです。

表 6-8 サンプリング補正処理

項目	説明		
概要	指定した周期に合わせて、以下の補正を行います。 ・リサンプリング：指定した周期にてファイルに数値がある場合、その数値を抜き出す。 ・データ補間：指定した周期にてファイルに数値がない場合、前後の数値をもとに、その時刻の数値を算出して補間する。		
要素説明	名称	値	説明
	type	linear	データ補間方法です。線形補間（linear）を指定してください。
	rate	正の実数値	サンプリングの周期を秒で指定してください。

定義例	例） サンプリング周期 10 秒でサンプリング補正
	{
	"method":"sampling",
	"param":{"type":"linear","rate":10}
	}		

備考			

6.5 統計情報ファイル

統計情報ファイルのフォーマットは、『図 6-7』の通りです。

```
{
        "channels":[
                {
                        "target":1,
                        "std":<1 列目の標準偏差値>,
                        "mean":<1 列目の平均値>,
                        "src_range":[
                                <1 列目の最小値>,
                                <1 列目の最大値>
                        ]
                },
                {
                        "target":2,
                        "std":<2 列目の標準偏差値>,
                        "mean":<2 列目の平均値>,
                        "src_range":[
                                <2 列目の最小値>,
                                <2 列目の最大値>
                        ]
                },
                …
         ],
         "all":{
        "total":<対象サンプル数総計>,
                "std":<全体の標準偏差値>,
                "mean":<全体の平均値>,
                "src_range":[
                        <全体の最小値>,
                        <全体の最大値>
                ]
        }
}
```

図 6-7 統計情報ファイルのフォーマット

6.6 コンフィグファイル

6.6.1 実行コンフィグファイル

実行コンフィグファイルの概要は『表 6-9』の通りです。

表 6-9 実行コンフィグファイルの概要

項目	説明
ファイル名	tsa_exec.conf
フォルダ/ディレクトリ名	[Linux 環境の場合] /opt/nec/rapid/etc/TSA

	[Windows 環境の場合] C:¥Users¥<user 名>¥AppData¥Local¥NEC¥Rapid¥TSA¥settings
文字コード	UTF-8（BOM なし）

実行コンフィグファイルの設定項目は、『表 6-10』の通りです。

表 6-10 実行コンフィグファイル設定項目

項目名	説明	データ型	範囲	デフォルト値
ENABLE_MULTI_THREAD	処理にマルチスレッドを使用するか否かのフラグです。	BOOL	true/false	true
MAX_LABEL_LENGTH	ラベル名の最大長です。	整数値	1-255	100
MAX_HEADER_LENGTH	時系列数値データファイルのヘッダ情報に記述する要素名の最大長です。	整数値	1-255	100
PROGRESS_OUT_INTERVAL	コマンド実行時の標準出力の時間間隔です。単位は秒です。	整数値	1-3600	5

実行コンフィグファイル記載例は『図 6-8』となります。

```
[ENABLE_MULTI_THREAD]=true
[MAX_LABEL_LENGTH]=100
[MAX_HEADER_LENGTH]=100
[PROGRESS_OUT_INTERVAL]=5
```

図 6-8 実行コンフィグファイルの記載例

⚠注意 実行コンフィグファイルの更新

実行コンフィグファイルの更新は、管理者ユーザで行ってください。

⚠注意 [ENABLE_MULTI_THREAD]の指定について

[ENABLE_MULTI_THREAD]に true を指定しても、期待通りに処理速度が改善されない場合があります。
また、[ENABLE_MULTI_THREAD]に true を指定した場合に、false を指定した場合に比べて処理速度が極端に遅くなるときは、以下の設定を調整して下さい。

対象ファイル名: <インストールパス>/template/TSA/lib/define.lua
対象パラメータ名: TOTAL_THREAD_NUM
調整の方針: お使いのマシンの論理コア数より小さい 1 以上の整数値を設定してください。

※論理コア数の確認方法
・Red Hat Enterprise Linux
以下のコマンドを実行してください。"processor"と出力された行数が論理コア数です。

```
less /proc/cpuinfo | grep processor
```

・Windows
コマンドプロンプトを起動し、以下のコマンドを実行してください。出力された数が論理コア数です。

```
set NUMBER_OF_PROCESSORS
```

6.6.2 学習コンフィグファイル

学習コンフィグファイルの概要は『表 6-11』の通りです。

表 6-11 学習コンフィグファイルの概要

項目	説明
ファイル名	tsa_train.conf
フォルダ/ディレクトリ名	[Linux 環境の場合] /opt/nec/rapid/etc/TSA [Windows 環境の場合] C:¥Users¥<user 名>¥AppData¥Local¥NEC¥Rapid¥TSA¥settings
文字コード	UTF-8（BOM なし）

学習コンフィグファイルの設定項目は『表 6-12』の通りです。

表 6-12 学習コンフィグファイル設定項目

項目名	説明	データ型	範囲	デフォルト値
共通設定				
TRAIN_RATE_ST	学習率の初期値です。	実数値	0-1	0.01
TRAIN_RATE_EN	学習率の最終値です。TRAIN_RATE_ST よりも小さい値を設定してください。	実数値	0-1	0.0001

TRAIN_EPOCH	学習の反復回数（epoch 数）です。	整数値	1-999999	30
TRAIN_SET_ITERATION	1epoch ごとの学習処理の繰り返し回数です。	整数値	1-999999	1
PROCESSOR	動作モードを指定してください。「cpu」または「gpu」が指定できます。	文字列	“cpu”/”gpu”	“cpu”
WINDOW_SIZE	ウィンドウサイズ（一度に抜き出して入力するデータの行数）です。1DOCN では、入力区間のサイズに相当します。入力区間については、『図 6-11』を参照してください。	整数値	15-9999	64
SLIDE_SIZE	ウィンドウをスライドさせる行数です。	整数値	1-9999	1
1DCNN 設定				
1DCNN_CHANNEL_INIT_SIZE	1DCNN 内のネットワーク規模の基準値です。大きい値を指定すると学習にかかる時間が増加します。	整数値	1-9999	64
1DCNN_CHANNEL_INC_SIZE	1DCNN 内のネットワーク規模の基準値からの増加量です。大きい値を指定すると学習にかかる時間が増加します。	整数値	0-9999	0
1DCNN_BATCH_SIZE	1DCNN のバッチサイズを指定します。	整数値	1-256	1
1DDCN 設定				
1DDCN_CHANNEL_INIT_SIZE	1DDCN 内のネットワーク規模の基準値です。大きい値を指定すると学習にかかる時間が増加します。	整数値	1-9999	32
1DDCN_CHANNEL_INC_SIZE	1DDCN 内のネットワーク規模の基準値からの増加量です。大きい値を指定すると学習にかかる時間が増加します。	整数値	0-9999	32
1DDCN_BATCH_SIZE	1DDCN のバッチサイズを指定します。	整数値	1-256	32
1DOCN 設定				
1DOCN_CHANNEL_INIT_SIZE	1DOCN 内のネットワーク規模の基準値です。大きい値を指定すると学習にかかる時間が増加します。	整数値	1-9999	16

1DOCN_CHANNEL_INC_SIZE	1DOCN 内のネットワーク規模の基準値からの増加量です。大きい値を指定すると学習にかかる時間が増加します。	整数値	0-9999	16
1DOCN_BATCH_SIZE	1DOCN のバッチサイズを指定します。	整数値	1-256	64
1DOCN_PREDICT_WINDOW_SIZE	1DOCN の判断区間のサイズ（データ行数）を指定します。判断区間については、『図 6-11』を参照してください。	整数値	8-9999	8

⚠注意 旧バージョンから変更になった設定項目について

学習コンフィグファイルの一部の設定項目は、旧バージョンから名称、デフォルト値が変更となっています。

表 6-13 旧バージョンから名称またはデフォルト値に変更のある設定項目

項目名		デフォルト値	
V1.2	V2.2	V1.2	V2.2
TRAIN_RATE_ST		0.1	0.01
EPOCH		100	30
WINDOW_SIZE		100	64
CHANNEL_INIT_SIZE	1DCNN_CHANNEL_INIT_SIZE	60	64
CHANNEL_INC_SIZE	1DCNN_CHANNEL_INC_SIZE	0	

⚠注意 1DOCN におけるパラメータ指定について

1DOCN の場合、学習データとして、入力区間に加え、それに続く判断区間が必要となります。入力区間、判断区間のサイズは、それぞれ、学習コンフィグファイルの[WINDOW_SIZE]、[1DOCN_PREDICT_WINDOW_SIZE]になります。入力区間と判断区間の関係については、『図 6-11』を参照してください。

[WINDOW_SIZE]や[1DOCN_PREDICT_WINDOW_SIZE]は、規定範囲の任意の整数値を指定することができますが、アルゴリズムの特性上、2 の累乗の値が推奨値となります。2 の累乗の値以外を指定した場合、内部でデータ補間処理が行われるため、精度が劣化する場合があります。

また、アルゴリズムの特性上、[1DOCN_PREDICT_WINDOW_SIZE]には、[WINDOW_SIZE]に指定した値以下の値を指定してください。

学習コンフィグファイルの記載例は『図 6-9』となります。

```
# common
[TRAIN_RATE_ST]=0.01
[TRAIN_RATE_EN]=0.0001
[TRAIN_EPOCH]=30
[TRAIN_SET_ITERATION]=1
```

```
[PROCESSOR]="cpu"
[WINDOW_SIZE]=64
[SLIDE_SIZE]=1

# 1DCNN
[1DCNN_CHANNEL_INIT_SIZE]=64
[1DCNN_CHANNEL_INC_SIZE]=0
[1DCNN_BATCH_SIZE]=1

# 1DDCN
[1DDCN_CHANNEL_INIT_SIZE]=32
[1DDCN_CHANNEL_INC_SIZE]=32
[1DDCN_BATCH_SIZE]=32

# 1DOCN
[1DOCN_CHANNEL_INIT_SIZE]=16
[1DOCN_CHANNEL_INC_SIZE]=16
[1DOCN_BATCH_SIZE]=64
[1DOCN_PREDICT_WINDOW_SIZE]=8
```

図 6-9 学習コンフィグファイルの記載例

⚠注意 -t で指定した学習アルゴリズム以外の設定項目について

学習コンフィグファイルについて、-t で指定した学習アルゴリズム以外の設定項目であっても、設定に誤りがある場合は、エラーとなります。
また、各設定項目の値が記述されていない場合は、警告メッセージが出力され、デフォルトのコンフィグファイルの値で動きます。
この警告メッセージは、-t で指定した学習アルゴリズム以外の項目が記述されていない場合でも出力されます。

WINDOWS_SIZE および SLIDE_SIZE と学習対象の関係は、『図 6-10』の通りです。

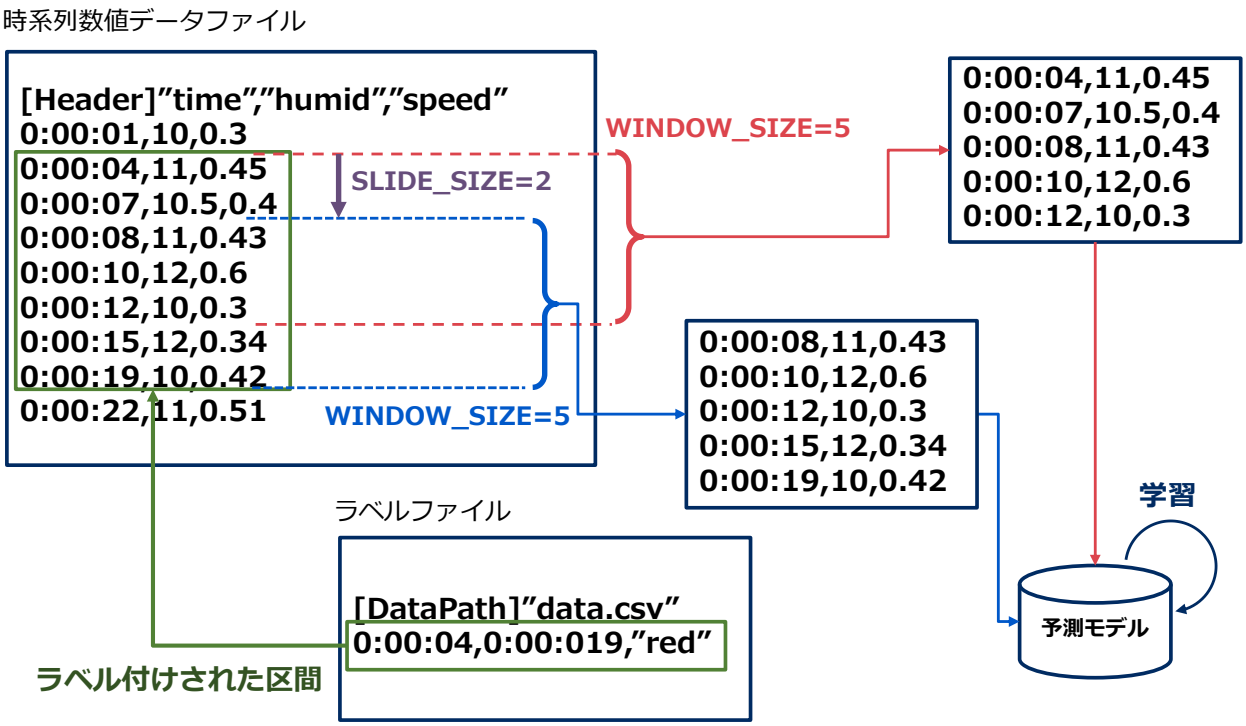

図 6-10 WINDOWS_SIZE および SLIDE_SIZE と学習対象の関係

OneClass 分類における、入力区間と判断区間の関係は、『図 6-11』の通りです。OneClass 分類では、入力区間を基に、それに続く判断区間を使って学習や予測を行うため、通常の入力区間に加え、判断区間のデータを抜き出して学習・予測を行います。入力区間、判断区間のサイズは、それぞれ学習コンフィグファイルの[WINDOW_SIZE]、[1DOCN_PREDICT_WINDOW_SIZE]で指定します。

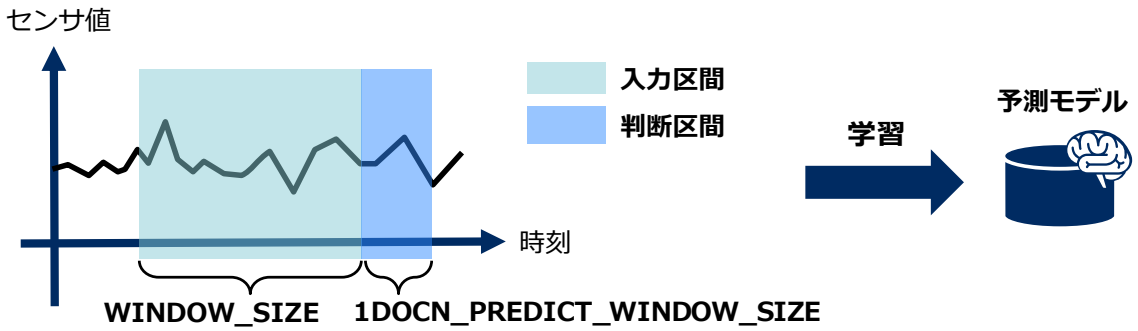

図 6-11 OneClass 分類における入力区間と判断区間の関係

6.6.3 予測コンフィグファイル

予測コンフィグファイルの概要は『表 6-14』の通りです。

表 6-14 予測コンフィグファイルの概要

項目	説明
ファイル名	tsa_predict.conf
フォルダ/ディレクトリ名	[Linux 環境の場合] /opt/nec/rapid/etc/TSA

	[Windows 環境の場合] C:¥Users¥<user 名>¥AppData¥Local¥NEC¥Rapid¥TSA¥settings
文字コード	UTF-8（BOM なし）

予測コンフィグファイルの設定項目は『表 6-15』の通りです。

表 6-15 予測コンフィグファイル設定項目

項目名	説明	データ型	範囲	デフォルト値
共通設定				
SLIDE_SIZE	ウィンドウをスライドさせる行数です。	整数値	1-9999	1
1DOCN 設定				
1DOCN_TOLERANCE	1DOCN の異常判定における、異常度に対する許容度を設定します。1DOCN の異常判定において、異常度がこの値を超えた場合に異常（NG）と判断します。	実数値	0-9999	1.0
1DOCN_FILTER_SIZE	1DOCN の異常判定における異常連続性フィルタのサイズを設定します。指定した回数分以上、連続で異常度が許容度を超えた場合に異常（NG）と判定します。それまでは、異常度が許容度を超えていても正常（OK）と判断します。	整数値	1-9999	1

予測コンフィグファイルの記載例は『図 6-12』となります。

```
# common
[SLIDE_SIZE]=1

# 1DOCN
[1DOCN_TOLERANCE]=1.0
[1DOCN_FILTER_SIZE]=1
```

図 6-12 予測コンフィグファイルの記載例

予測モデルの WINDOW_SIZE および SLIDE_SIZE と予測対象の関係は『図 6-13』の通りです。

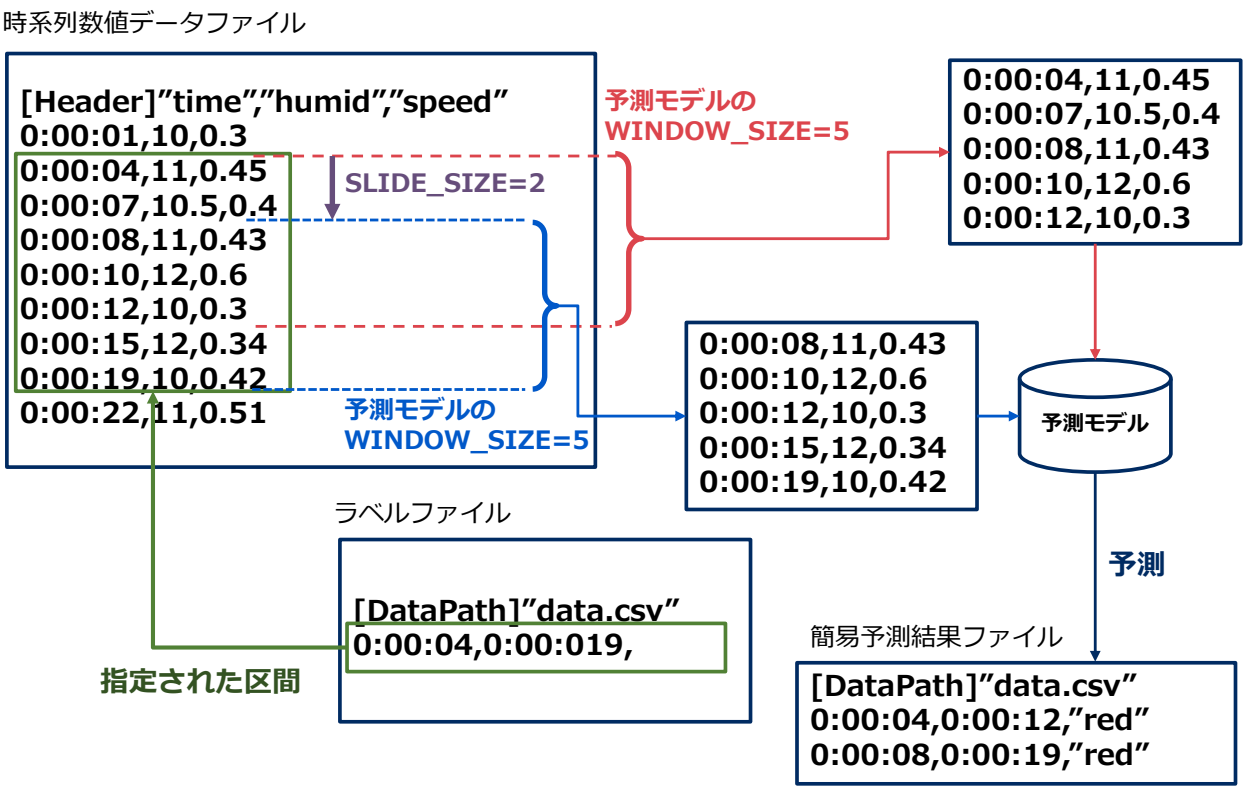

図 6-13 予測モデルの WINDOW_SIZE および SLIDE_SIZE と予測対象の関係

OneClass 分類の異常判定における異常連続性フィルタについて

OneClass 分類の予測では、予測モデルで求めた異常度が許容度を超えた箇所を異常箇所と判断しますが、ノイズ等による一時的な外れ値を異常箇所と誤判定する場合があります。これを回避する方法として、異常連続性フィルタがあります。

異常連続性フィルタの概要を『図 6-14』に記載します。

異常連続性フィルタは、サイズ([1DOCN_FILTER_SIZE])として設定された値の回数分以上、連続で異常度が許容度を超えた場合に異常と判断し、それ以外は正常と判断します。デフォルトでは、フィルタのサイズには 1 が設定されており、異常連続性フィルタは無効の状態となっています。フィルタのサイズに 2 以上の値を指定すると、異常連続性フィルタが有効となり、外れ値によって一時的に異常度が許容度を超えても、フィルタのサイズ未満の連続回数であれば正常と判断し、異常と誤判定するのを回避することができます。

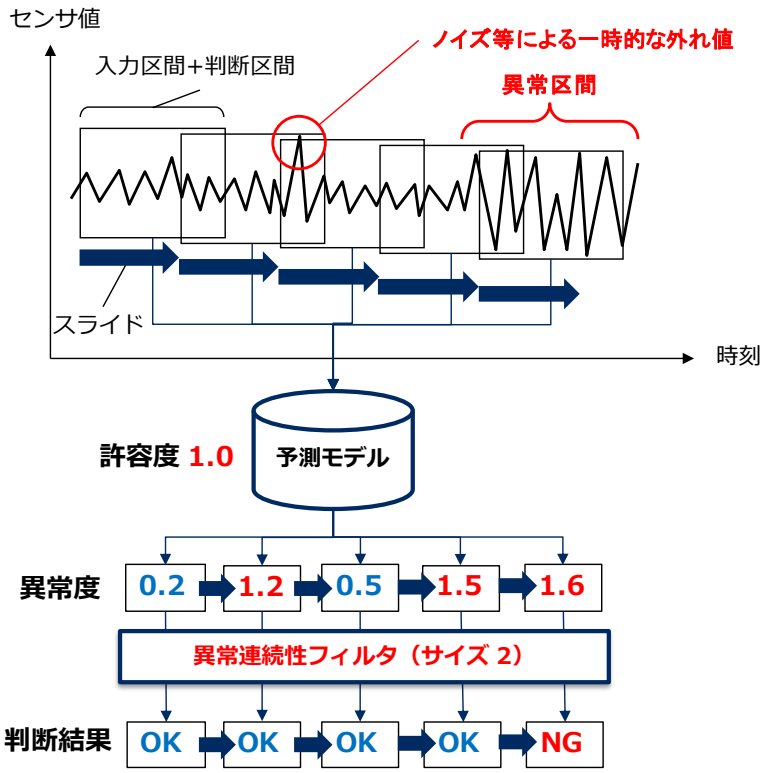

図 6-14 OneClass 分類の異常連続性フィルタの概要

6.6.4 オンライン予測コンフィグファイル

オンライン予測コンフィグファイルの概要は『表 6-16』の通りです。

表 6-16 オンライン予測コンフィグファイルの概要

項目	説明
ファイル名	tsa_socket_daemon.conf
フォルダ/ディレクトリ名	[Linux 環境の場合] /opt/nec/rapid/etc/TSA [Windows 環境の場合] C:¥Users¥<user 名>¥AppData¥Local¥NEC¥Rapid¥TSA¥settings
文字コード	UTF-8（BOM なし）

オンライン予測コンフィグファイルの設定項目は『表 6-17』の通りです。

表 6-17 オンライン予測コンフィグファイル設定項目

項目名	説明	データ型	範囲	デフォルト値
TIMEOUT	TCP ソケットの接続待ちタイムアウト(秒)を指定します。0 を指定した場合はタイムアウト処理を行いません。	整数値	0-3600	180

MODEL_DIR	予測に用いるモデルファイルが保存されたディレクトリの親ディレクトリパスを設定します。ディレクトリパスはダブルクォートで括ってください。 ※環境変数は使用できません。 省略した場合や値を指定しない場合は以下を参照します。 [Linux] /opt/nec/rapid/etc/TSA/models [Windows] C:¥ProgramData¥NEC¥Rapid¥TSA¥models	絶対パス	規定なし	<指定なし>
CACHED_MODEL	常駐プロセスの起動時にキャッシュするモデルファイルの保存されたディレクトリ名のリストを設定します。ダブルクォートで括り、ディレクトリ名ごとにカンマ(,)区切りで指定してください。カンマの前後には、実際のディレクトリ名に含まれないスペースは使用しないでください。値を指定しない場合は、モデルをキャッシュしない状態で起動します。 例） [CACHED_MODEL]="model1,model2,model3" 下記ディレクトリ名のモデルをキャッシュします。 「model1」 「model2」 「model3」	文字列	64 個以下のディレクトリ名	<指定なし>
MODEL_CACHE_SIZE	常駐プロセスにてキャッシュするモデルの最大数を設定します。この値を超えた場合は最初に読み込んだモデルから順にキャッシュから削除されます。	整数値	1-9999	64

オンライン予測コンフィグファイルの記載例は『図 6-15』となります。

```
[TIMEOUT]=180
[MODEL_DIR]="C:¥Users¥user01¥AppData¥Local¥NEC¥Rapid¥TSA¥models"
[CACHED_MODEL]="model01,model02,model03"
[MODEL_CACHE_SIZE]=64
```

図 6-15 オンライン予測コンフィグファイルの記載例

6.6.5 CLI ログコンフィグファイル

CLI ログコンフィグファイルの概要は『表 6-18』の通りです。

表 6-18 CLI ログコンフィグファイルの概要

項目	説明
ファイル名	logsetting.conf
フォルダ/ディレクトリ名	[Linux 環境の場合] /opt/nec/rapid/etc/TSA [Windows 環境の場合] C:¥ProgramData¥NEC¥Rapid¥TSA
文字コード	UTF-8（BOM なし）

CLI ログコンフィグファイルの設定項目は『表 6-19』の通りです。

表 6-19 CLI ログコンフィグファイル設定項目

項目名	説明	データ型	範囲	デフォルト値
RAPID_LOGGER_PATH	ログ出力先ディレクトリのパスです。マルチバイト文字の指定もできます。	絶対パス	規定なし	[Linux 環境の場合] /var/log/nec/rapid [Windows 環境の場合] %USERPROFILE%¥AppData¥Local¥NEC¥Rapid¥log
RAPID_LOGGER_SIZE	1 世代あたりのログファイルのデータサイズです。 最新世代のログファイルのサイズが本項目の値を超えた場合、ログファイルのローテーションが発生します。	整数値	1-999999999	10000000
RAPID_LOGGER_ROTATE	ログファイルの最大世代数です。 ログメッセージを書き込むファイルも 1 世代分としてカウントします。	整数値	1-9	3

RAPID_LOGGER_OUTLEVEL	ログ出力レベルです。指定したレベル以上のログがファイルに出力されます。	文字列	DEBUG/ INFO/ WARN/ ERROR/ FATAL	INFO

CLI ログコンフィグファイルの記載例は『図 6-16』となります。

```
[RAPID_LOGGER_PATH]=/var/log/nec/rapid
[RAPID_LOGGER_SIZE]=10000000
[RAPID_LOGGER_ROTATE]=3
[RAPID_LOGGER_OUTLEVEL]=INFO
```

図 6-16 CLI ログコンフィグファイルの記載例(Red Hat Enterprise Linux)

6.6.6 MLE ログコンフィグファイル

MLE ログコンフィグファイルの概要は『表 6-20』の通りです。

表 6-20 MLE ログコンフィグファイルの概要

項目	説明
ファイル名	logsetting.conf
フォルダ/ディレクトリ名	[Linux 環境の場合] /opt/nec/rapid/etc/MLE [Windows 環境の場合] C:¥ProgramData¥NEC¥Rapid¥MLE
文字コード	UTF-8（BOM なし）

MLE ログコンフィグファイルの設定項目は『表 6-21』の通りです。

表 6-21 MLE ログコンフィグファイル設定項目

項目名	説明	データ型	範囲	デフォルト値
RAPID_LOGGER_PATH	ログ出力先ディレクトリのパスです。マルチバイト文字の指定もできます。	絶対パス	規定なし	[Linux 環境の場合] /var/log/nec/rapid [Windows 環境の場合] %ProgramData%¥NEC¥Rapid¥log
RAPID_LOGGER_SIZE	1 世代あたりのログファイルのデータサイズです。 最新世代のログファイルのサイズが本項目の値を超えた場合、ログファイルのローテーションが発生します。	整数値	1-999999999	10000000
RAPID_LOGGER_ROTATE	ログファイルの最大世代数です。 ログメッセージを書き込むファイルも 1 世代分としてカウントします。	整数値	1-9	3
RAPID_LOGGER_OUTLEVEL	ログ出力レベルです。指定したレベル以上のログがファイルに出力されます。	文字列	DEBUG/ INFO/ WARN/ ERROR/ FATAL	INFO

MLE ログコンフィグファイルの記載例は『図 6-17』となります。

```
[RAPID_LOGGER_PATH]=/var/log/nec/rapid
[RAPID_LOGGER_SIZE]=10000000
[RAPID_LOGGER_ROTATE]=3
[RAPID_LOGGER_OUTLEVEL]=INFO
```

図 6-17 MLE ログコンフィグファイルの記載例(Red Hat Enterprise Linux)

6.7 ログファイル

本製品で出力されるログファイルには CLI ログファイルと MLE ログファイルがあります。ログファイルはそれぞれのログコンフィグファイルで指定したフォルダ/ディレクトリに出力されます。

本製品で出力されるログファイルの形式は『表 6-22』の通りです。

表 6-22 ログファイル出力形式表

ログ出力先ディレクトリパス	CLI/MLE ログコンフィグファイルの[RAPID_LOGGER_PATH]で設定したディレクトリに出力されます。 CLI ログコンフィグファイルについては、以下が既定値になります。 [Linux 環境の場合] /var/log/nec/rapid [Windows 環境の場合] %USERPROFILE%¥AppData¥Local¥NEC¥Rapid¥log MLE ログコンフィグファイルについては、以下が規定値になります。 [Linux 環境の場合] /var/log/nec/rapid [Windows 環境の場合] %ProgramData%¥NEC¥Rapid¥log
ログファイル名	TSA_<世代数>.log MLE_<世代数>.log
ログフォーマット	<level>␣<pid>␣TSA␣<func_name>␣<date>␣<time>␣<message> <level>␣<pid>␣MLE␣<date>␣<time>␣<message>
項目名	**内容**
level	ログレベルを出力します。 DEBUG/INFO/WARN/ERROR/FATAL のうちいずれかとなります。
pid	RAPID プロセスの ID です。
func_name	ログを出力している機能の名前です。
date	ログを出力した日付です。yyyy-mm-dd のフォーマットで出力します。
time	ログを出力した時刻です。hh:mm:ss.sss のフォーマットで出力します。
message	ログメッセージです。
出力レベル	**説明**
DEBUG	動作確認用の情報です。
INFO	RAPID プロセスの情報です。
WARN	警告です。

ERROR	エラーです。
FATAL	重大なエラーです。

第7章 注意・制限事項

本章では、本製品の注意事項、制限事項について説明します。

7.1 注意事項

7.1.1 本製品の使用時に使用するユーザ権限

- 本製品は、標準ユーザで使用してください。管理者ユーザでは使用しないでください。

7.1.2 実行コンフィグファイルの更新に使用するユーザ権限

- 実行コンフィグファイルの更新は、管理者ユーザで行ってください。

7.1.3 前処理コマンド実行後の学習・予測コマンドの実行について

- –o オプションを指定せず前処理コマンドを実行した後、ディレクトリを指定して学習・予測を行う場合、前処理の結果出力された前処理済み時系列数値データファイルとラベルファイルを別ディレクトリに移動してから、実行してください。

7.1.4 1DOCN の学習における処理時間について

- 1DOCN を指定して学習実行する場合、進捗状況に"TRAIN"のステータス処理 100%完了の旨が出力された後、次の進捗が出力されるまでに1エポック分の学習と同程度の時間を要します。

7.1.5 学習実行中のコンフィグファイルの編集について

- 学習プロセスを立ち上げている最中に、学習に使用しているコンフィグファイル（学習コンフィグファイル、前処理定義ファイル）の内容を編集しないでください。
  学習の途中でコンフィグファイルの編集を行うと、予測モデル保存時に出力されるファイル間に不整合が生じ、予測が正しく実行できなくなります。

7.2 制限事項

なし

NEC Advanced Analytics - RAPID 機械学習
時系列数値解析 V2.2
ユーザガイド

2018 年 10 月

©2018 NEC Corporation