IMADICS: IPFIX Monitoring and Anomaly Detection in Industrial Control Systems

1.0

Generated by Doxygen 1.9.5

1 Namespace Documentation

1.1 IcsMonitor Namespace Reference

1.2 IcsMonitor.AnomalyDetection Namespace Reference

Classes

- class ClusterModel

  Represents the K-means-based anomaly detection model.
  The model consists of a set of clusters each complemented with its variance.Each cluster thus represents a sphere in the n-dimensional space.If a communication pattern characterized by a point in the space belongs to some sphere, it is marked as normal. Otherwise, it is anomalous.
- interface IAnomalyDetectionModel

  Defines common interface for anomaly detection models.
- class ModelTrainer

  Represents anomaly detection model trainer.
  It provides methods for training different anomaly detection methods.
- class TrafficProfile

  Represents traffic profile that consists of a collection of models. The profile is used for anomaly detection provided the network traffic.
- class TrafficProfileTrainer

  Trainer for creating a profile based on the provided dataview.

Enumerations

- enum IndustrialProtocol

  A collection of currently supported industrial protocols.

Functions

- record FlowScore (string FlowKey, DateTime WindowStart, TimeSpan WindowDuration, string FlowLabel, float[ ] Features, double[ ] Distances, double[ ] Scores)

  Represents the score of the each flow as computed by the profile.

1.2.1 Function Documentation

1.2.1.1 FlowScore() `record IcsMonitor.AnomalyDetection.FlowScore (`

```
string FlowKey,
DateTime WindowStart,
TimeSpan WindowDuration,
string FlowLabel,
float[] Features,
double[] Distances,
double[] Scores )
```

Represents the score of the each flow as computed by the profile.

Parameters

FlowKey	The flow key.
Features	Values of the computed features.
Distances	An array of distances to the closes centroids for all models.
Scores	An array of scores computed for each model.

Gets the maximum score.

Gets the minimum score.

Gets the averegae score.

Gets the index of the best model, i.e., a model having the best score.

1.3 IcsMonitor.Flows Namespace Reference

Classes

- class AggregatorKey

  This static class provides different aggregation keys.
- class FlowsDataViewSource

  An abstract class that also provides a method to create specific flow sources for different supported protocols.
- class PacketAnnotationSourceFile

  Represents a packet annotation source file.
  Packet annotation is a CSV file that matches labesl to packet numbers.
- class PacketDeviceSource

  Supports observable for the capture device.

Functions

- record MultiflowKey (System.Net.Sockets.ProtocolType ProtocolType, IPAddress ClientIpAddress, IPAddress ServerIpAddress, ushort ServerPort)

  Represents a multiflow key used to aggregate records. This is the compound key. It aggreagtes flows to multiflow such that all flows in the bag have the same protocol types, client address, server address and the server port.
- record BiflowKey (System.Net.Sockets.ProtocolType ProtocolType, IPAddress ClientIpAddress, ushort ClientPort, IPAddress ServerIpAddress, ushort ServerPort)

  Represents a biflow key used to aggregate the flow records. It aggregates the flows of bidirectional conversations.
- record FlowMetrics (int Packets, long Octets, long FirstSeen, long LastSeen)

  A record of basic flow metrics.
- record PacketRecord< TPacket > (long Ticks, string Label, FlowKey Key, TPacket Packet)

  Represents a single parsed packet.

1.3.1 Function Documentation

1.3.1.1 BiflowKey() `record IcsMonitor.Flows.BiflowKey (`

```
System.Net.Sockets.ProtocolType ProtocolType,
IPAddress ClientIpAddress,
ushort ClientPort,
IPAddress ServerIpAddress,
ushort ServerPort )
```

Represents a biflow key used to aggregate the flow records. It aggregates the flows of bidirectional conversations.

Parameters

ProtocolType	the protocol type.
ClientIpAddress	The client address.
ClientPort	The client port.
ServerIpAddress	The server address.
ServerPort	The server port.

1.3.1.2 FlowMetrics() `record IcsMonitor.Flows.FlowMetrics (`

```
int Packets,
long Octets,
long FirstSeen,
long LastSeen )
```

A record of basic flow metrics.

Parameters

Packets	Number of packet of the flow.
Octets	Number of octets of the flow.
FirstSeen	Timestamp of observed first packet of the flow.
LastSeen	Timestamp of observed last packet of the flow.

The start of the flow as DateTime value.

The duration of the flow as TimeSpan value.

Aggregates two flow metrics.

Parameters

x	The flow metrics.
y	The flow metrics.

Returns

Aggregated flow metrics.

Gets the lower from the two provide ticks if they are greater than zero. Zero stands for invalid/undefined value.

Parameters

x	The first tick value.
y	the second tick value.

Returns

The smallest of the provided tick values.

Gets the greater from the two provide ticks. Zero stands for invalid/undefined value.

Parameters

x	The first tick value.
y	the second tick value.

Returns

The greatest of the provided tick values.

1.3.1.3 MultiflowKey()

```
record IcsMonitor.Flows.MultiflowKey (
        System.Net.Sockets.ProtocolType ProtocolType,
        IPAddress ClientIpAddress,
        IPAddress ServerIpAddress,
        ushort ServerPort )
```

Represents a multiflow key used to aggregate records. This is the compound key. It aggreagtes flows to multiflow such that all flows in the bag have the same protocol types, client address, server address and the server port.

Parameters

ProtocolType	The protocol type.
ClientIpAddress	The client address.
ServerIpAddress	The server address.
ServerPort	The server port.

1.3.1.4 PacketRecord< TPacket >()

```
record IcsMonitor.Flows.PacketRecord< TPacket > (
    long Ticks,
    string Label,
    FlowKey Key,
    TPacket Packet )
```

Represents a single parsed packet.

Parameters

Ticks	Packet timestamp in ticks resolution.
Key	The flow key of the packet.
Packet	The packet data.

Creates a packet record from the raw capture.

Parameters

rawCapture	The raw capture of the packet.
label	The label associated with the packet (if any).

Returns

The packet record for the capture.

Gets the packet timestamp as DateTime struct.

1.4 IcsMonitor.Modbus Namespace Reference

Classes

- class ModbusCompact

  A compact version that only counts number of operations of each operation type.
- class ModbusDataViewRecord

  Represents a flattened record used as a typed version for corresponding Dataviews.
  This class is computed from Flows.FlowRecord<ModbusCompact> and can be used for accesing dataview records.
- class ModbusDataViewSource

  An implementation of data view source for MODBUS protocol.
- class ModbusFlowProcessor

  Flow processor for extracting MODBUS related information from bidirectional flows.
- struct ModbusRawData

  A full version of the MODBUS flow record.

1.5 IcsMonitor.Protocols Namespace Reference

Classes

- class IecDataViewRecord

  This class represents the IEC record as used in ML's DataView.
- class IecDataViewRecordFlowmon

  Represents IEC IPFIX record as defined by Flowmon. It is loaded by CsvHelper and thus its properties need to be annotated with
  See also

  CsvName

  attribute.
- class IecDataViewRecordWireshark

  Represents IEC IPFIX record as produced by Wireshark IEC dissector.

1.6 IcsMonitor.Utils Namespace Reference

Classes

- class OptionHelper

  A helper class for processing command line options.
- class ZipEntryYamlIO

  An extension class for I/O operations with ZipArchiveEntry.

1.7 Traffix Namespace Reference

1.8 Traffix.DataView Namespace Reference

Classes

- class DataViewSaverCatalog

  Collection of extension methods for the DataOperationsCatalog to wrote to various text files such as csv, yaml, md and json.
- class DataViewWriterBase

  Abstract base class for custom data view writers.
- class DataViewWriterFactory

  The fatory for providing writers of the supported file formats.
- interface IDataViewWriter

  A common interface for implementations of data view writers.
- class TraffixTransformsCatalog

  The extension class implementing project's specific transformers.

Enumerations

- enum OutputFormat

  Defines supported data output formats.

2 Class Documentation

2.1 IcsMonitor.Flows.AggregatorKey Class Reference

This static class provides different aggregation keys.

Static Public Member Functions

- static MultiflowKey Multiflow (FlowKey arg)

  This key aggregates all flows between a client endpoint (any client port) and the server socket endpoint.
- static BiflowKey Biflow (FlowKey arg)

  This key aggregates all flows between the client socket endpoint and the server socket endpoint. It corresponds the bidirectional flow (conversation).

2.1.1 Detailed Description

This static class provides different aggregation keys.

2.1.2 Member Function Documentation

2.1.2.1 Biflow() `static BiflowKey IcsMonitor.Flows.AggregatorKey.Biflow ( FlowKey arg ) [inline], [static]`

This key aggregates all flows between the client socket endpoint and the server socket endpoint. It corresponds the bidirectional flow (conversation).

Parameters

arg	The flow key.

Returns

The aggregation key.

2.1.2.2 Multiflow() `static MultiflowKey IcsMonitor.Flows.AggregatorKey.Multiflow ( FlowKey arg ) [inline], [static]`

This key aggregates all flows between a client endpoint (any client port) and the server socket endpoint.

Parameters

arg	The flow key.

Returns

The aggregation key.

The documentation for this class was generated from the following file:

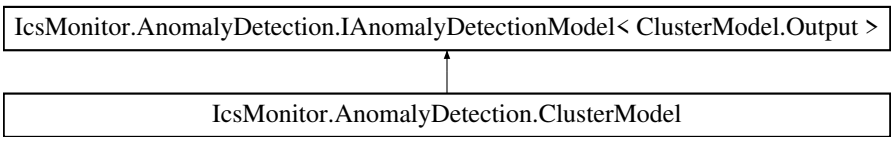
- Flows/AggregatorKey.cs

2.2 IcsMonitor.AnomalyDetection.ClusterModel Class Reference

Represents the K-means-based anomaly detection model.

The model consists of a set of clusters each complemented with its variance.Each cluster thus represents a sphere in the n-dimensional space.If a communication pattern characterized by a point in the space belongs to some sphere, it is marked as normal. Otherwise, it is anomalous.

Inheritance diagram for IcsMonitor.AnomalyDetection.ClusterModel:



Classes

- class Options

  Defines the options for creating the model.
- class Output

  Represents the cluster prediction data type. This is the output type from the prediction.
  See tutorial on K-Means clustering for more details: https://docs.microsoft.com/en-us/dotnet/machine-learni

Public Member Functions

- void **SaveToFile** (MLContext mlContext, string path)
- void Save (MLContext mlContext, ZipArchive archive, string prefix)

  Saves the model to the given Zip archive.
- IEnumerable< Output > **Transform** (MLContext mlContext, IDataView source)
- ClusteringMetrics Evaluate (MLContext mlContext, IDataView testData)

  Computes metrics by evaluating the model for the given input data.

Static Public Member Functions

- static ClusterModel LoadFromFile (MLContext mlContext, string path)

  Loads the model from the given file.
- static ClusterModel Load (MLContext mlContext, ZipArchive modelArchive, string prefix)

  Loads the model from the given archive.

Properties

- float[ ][ ] **Centroids** `[get]`

  Gets coordinates of centroids.
- float[ ] **Variances** `[get]`

  Gets variances of clusters.

2.2.1 Detailed Description

Represents the K-means-based anomaly detection model.

The model consists of a set of clusters each complemented with its variance.Each cluster thus represents a sphere in the n-dimensional space.If a communication pattern characterized by a point in the space belongs to some sphere, it is marked as normal. Otherwise, it is anomalous.

2.2.2 Member Function Documentation

2.2.2.1 Evaluate()

```
ClusteringMetrics IcsMonitor.AnomalyDetection.ClusterModel.Evaluate (
        MLContext mlContext,
        IDataView testData )  [inline]
```

Computes metrics by evaluating the model for the given input data.

Parameters

mlContext	The ML.NET context.
testData	Test data used for metrics computation.

Returns

2.2.2.2 Load()

```
static ClusterModel IcsMonitor.AnomalyDetection.ClusterModel.Load (
        MLContext mlContext,
        ZipArchive modelArchive,
        string prefix )  [inline], [static]
```

Loads the model from the given archive.

Parameters

mlContext	The ML.NET context.
modelArchive	Zip archive to read data from.
prefix	The prefix of entries in the ZIP archive.

Returns

The loaded model.

2.2.2.3 LoadFromFile()

```
static ClusterModel IcsMonitor.AnomalyDetection.ClusterModel.LoadFrom↩
File (
            MLContext mlContext,
            string path )  [inline], [static]
```

Loads the model from the given file.

Parameters

mlContext	The ML context.
path	Path to the model file.

Returns

The new anomaly detection model.

2.2.2.4 Save()

```
void IcsMonitor.AnomalyDetection.ClusterModel.Save (
            MLContext mlContext,
            ZipArchive archive,
            string prefix )  [inline]
```

Saves the model to the given Zip archive.

Parameters

mlContext	The ML.NET context.
archive	The Zip archive to save the model to.
prefix	The prefix used for naming the entries in the Zip archives.

The documentation for this class was generated from the following file:

- AnomalyDetection/ClusterModel.cs

2.3 Traffix.DataView.DataViewSaverCatalog Class Reference

Collection of extension methods for the DataOperationsCatalog to wrote to various text files such as csv, yaml, md and json.

Static Public Member Functions

- static void SaveAsCsvText (this Microsoft.ML.DataOperationsCatalog _, Microsoft.ML.IDataView data, System.IO.Stream stream)

  Save the IDataView as CSV text.
- static void SaveAsJsonText (this Microsoft.ML.DataOperationsCatalog _, Microsoft.ML.IDataView data, Stream stream)

  Save the IDataView as JSON text.
- static void SaveAsMarkdownText (this Microsoft.ML.DataOperationsCatalog _, Microsoft.ML.IDataView data, System.IO.Stream stream)

  Save the IDataView as Markdown table.
- static void SaveAsYamlText (this Microsoft.ML.DataOperationsCatalog _, Microsoft.ML.IDataView data, System.IO.Stream stream)

  Save the IDataView as YAML table.

2.3.1 Detailed Description

Collection of extension methods for the DataOperationsCatalog to wrote to various text files such as csv, yaml, md and json.

2.3.2 Member Function Documentation

2.3.2.1 SaveAsCsvText()

```
static void Traffix.DataView.DataViewSaverCatalog.SaveAsCsvText (
    this Microsoft.ML.DataOperationsCatalog _,
    Microsoft.ML.IDataView data,
    System.IO.Stream stream )  [inline], [static]
```

Save the IDataView as CSV text.

Parameters

catalog	The DataOperationsCatalog catalog.
data	The data view to save.
stream	The stream to write to.

2.3.2.2 SaveAsJsonText()

```
static void Traffix.DataView.DataViewSaverCatalog.SaveAsJsonText (
    this Microsoft.ML.DataOperationsCatalog _,
    Microsoft.ML.IDataView data,
    Stream stream )  [inline], [static]
```

Save the IDataView as JSON text.

Parameters

catalog	The DataOperationsCatalog catalog.
data	The data view to save.
stream	The stream to write to.

2.3.2.3 SaveAsMarkdownText()

```
static void Traffix.DataView.DataViewSaverCatalog.SaveAsMarkdown↩
Text (
            this Microsoft.ML.DataOperationsCatalog _,
            Microsoft.ML.IDataView data,
            System.IO.Stream stream )  [inline], [static]
```

Save the IDataView as Markdown table.

Parameters

catalog	The DataOperationsCatalog catalog.
data	The data view to save.
stream	The stream to write to.

2.3.2.4 SaveAsYamlText()

```
static void Traffix.DataView.DataViewSaverCatalog.SaveAsYamlText (
            this Microsoft.ML.DataOperationsCatalog _,
            Microsoft.ML.IDataView data,
            System.IO.Stream stream )  [inline], [static]
```

Save the IDataView as YAML table.

Parameters

catalog	The DataOperationsCatalog catalog.
data	The data view to save.
stream	The stream to write to.

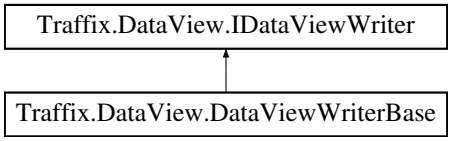
The documentation for this class was generated from the following file:

- DataView/DataViewSaverCatalog.cs

2.4 Traffix.DataView.DataViewWriterBase Class Reference

Abstract base class for custom data view writers.

Inheritance diagram for Traffix.DataView.DataViewWriterBase:



Public Member Functions

- void **Dispose** ()
- int AppendDataView (IDataView dataview)

  Can be used to append the dataview to the current writer.
- void **BeginDocument** ()

  Writes the beginning of the document.
- void **EndDocument** ()

  Writes the end of the document.

Protected Member Functions

- DataViewWriterBase (TextWriter writer, DataViewSchema schema)

  The constructor.
- virtual void Dispose (bool disposing)

  Implements the dispose pattern.
- abstract void **WriteHeader** ()

  Implement to write the specific header of the document.
- abstract void **WriteFooter** ()

  Implement to write the specific footer of the document.
- abstract void WriteRow (IEnumerable< KeyValuePair< string, object > > values)

  Implement to write a sinlge row/record of the document.
- virtual void **CleanUp** ()

  Called before the writer is closed and disposed.
- ExpandoObject GetExpandoScheme (IEnumerable< KeyValuePair< string, object > > values)

  Gets the expando object for the given key-value pairs.
- ExpandoObject GetExpandoObject (IEnumerable< KeyValuePair< string, object > > values)

  Gets the expando object for the given key-value pairs.

Static Protected Member Functions

- static object GetStringValueForColumn (DataViewSchema.Column column, DataViewRowCursor cursor)

  Gets the string value for the given column.
- static object GetVectorValue (DataViewRowCursor cursor, DataViewSchema.Column column, Primitive↩ DataViewType itemType)

  Gets the vector value for the given field in data view.
- static T GetValue< T > (DataViewRowCursor cursor, DataViewSchema.Column column)

  Gets the field value as an object of type T .
- static string GetTextValue (DataViewRowCursor cursor, DataViewSchema.Column column)

  Gets the field value as text (string).
- static IEnumerable< KeyValuePair< string, object > > GetValues (DataViewRowCursor cursor, DataView↩ Schema.Column[ ] columns)

  Gets the values of the given colleciton of columns.

Properties

- IndentedTextWriter **Writer** `[get]`

  Gets the indented writer used for writing the output.
- DataViewSchema **Schema** `[get]`

  Gets the associated data view schmema.
- DataViewSchema.Column[ ] **Columns** `[get]`

  Gets the collection of schema columns.

2.4.1 Detailed Description

Abstract base class for custom data view writers.

2.4.2 Constructor & Destructor Documentation

2.4.2.1 DataViewWriterBase() `Traffix.DataView.DataViewWriterBase.DataViewWriterBase (`

```
TextWriter writer,
DataViewSchema schema )  [inline], [protected]
```

The constructor.

Parameters

writer	The writer used to produce the output.
schema	The data view schema.

2.4.3 Member Function Documentation

2.4.3.1 AppendDataView() `int Traffix.DataView.DataViewWriterBase.AppendDataView (`

```
IDataView dataview )  [inline]
```

Can be used to append the dataview to the current writer.

Parameters

dataview	The dataview to append. It can be null if header or footer needs to be written.
writeHeader	true if header should be written before the dataview rows.
writeFooter	true if footer should be written after the dataview rows.

Implements Traffix.DataView.IDataViewWriter.

2.4.3.2 Dispose() `virtual void Traffix.DataView.DataViewWriterBase.Dispose (`

```
bool disposing )  [inline], [protected], [virtual]
```

Implements the dispose pattern.

Parameters

disposing	True if object is being disposed.

2.4.3.3 GetExpandoObject()

```
ExpandoObject Traffix.DataView.DataViewWriterBase.GetExpandoObject
(
            IEnumerable< KeyValuePair< string, object > > values )  [inline], [protected]
```

Gets the expando object for the given key-value pairs.

Parameters

values	Values with their column names.

Returns

The expando object for the given colleciton of key-values.

2.4.3.4 GetExpandoScheme()

```
ExpandoObject Traffix.DataView.DataViewWriterBase.GetExpandoScheme
(
            IEnumerable< KeyValuePair< string, object > > values )  [inline], [protected]
```

Gets the expando object for the given key-value pairs.

Parameters

values	Values with their column names.

Returns

The expando object for the given colleciton of key-values.

2.4.3.5 GetStringValueForColumn()

```
static object Traffix.DataView.DataViewWriterBase.GetString↩
ValueForColumn (
            DataViewSchema.Column column,
            DataViewRowCursor cursor )  [inline], [static], [protected]
```

Gets the string value for the given column.

Parameters

column	The data view column.
cursor	The cursor pointing to the actual row in the data view.

Returns

Exceptions

NotSupportedException	

2.4.3.6 GetTextValue() `static string Traffix.DataView.DataViewWriterBase.GetTextValue ( DataViewRowCursor cursor, DataViewSchema.Column column ) [inline], [static], [protected]`

Gets the field value as text (string).

Parameters

cursor	The row cursor.
column	The data view column.

Returns

The string representing the field value.

2.4.3.7 GetValue< T >() `static T Traffix.DataView.DataViewWriterBase.GetValue< T > ( DataViewRowCursor cursor, DataViewSchema.Column column ) [inline], [static], [protected]`

Gets the field value as an object of type *T*.

Template Parameters

T	The required object type.

Parameters

cursor	The row cursor.
column	The data view column.

Returns

The field value of type *T*.

2.4.3.8 GetValues()

```
static IEnumerable< KeyValuePair< string, object > > Traffix.DataView.↩
DataViewWriterBase.GetValues (
            DataViewRowCursor cursor,
            DataViewSchema.Column[] columns )  [inline], [static], [protected]
```

Gets the values of the given colleciton of columns.

Parameters

cursor	The row cursor.
columns	The colleciton of columns to retrieve.

Returns

The key-value pairs representing the values for the requested columns at the given cursor.

2.4.3.9 GetVectorValue()

```
static object Traffix.DataView.DataViewWriterBase.GetVectorValue (
            DataViewRowCursor cursor,
            DataViewSchema.Column column,
            PrimitiveDataViewType itemType )  [inline], [static], [protected]
```

Gets the vector value for the given field in data view.

Parameters

cursor	The row cursor.
column	the data view column.
itemType	The type of field.

Returns

The vector value representing the given field.

Exceptions

NotSupportedException	For types that are not representable as vector values.

2.4.3.10 WriteRow()

```
abstract void Traffix.DataView.DataViewWriterBase.WriteRow (
            IEnumerable< KeyValuePair< string, object > > values )  [protected], [pure virtual]
```

Implement to write a sinlge row/record of the document.

Parameters

values	

The documentation for this class was generated from the following file:

- DataView/DataViewWriterBase.cs

2.5 Traffix.DataView.DataViewWriterFactory Class Reference

The fatory for providing writers of the supported file formats.

Static Public Member Functions

- static IDataViewWriter CreateWriter (OutputFormat format, TextWriter writer, Microsoft.ML.DataViewSchema schema)

  Creates a writer of the request file format.
  Writer is created for the specific file format and data view schema. The data view of the compatible schema can be written using the created writer. It is because some formats, e.g., CSV needs to know the schema in advance in order to write the header.

2.5.1 Detailed Description

The fatory for providing writers of the supported file formats.

2.5.2 Member Function Documentation

2.5.2.1 CreateWriter() `static IDataViewWriter Traffix.DataView.DataViewWriterFactory.CreateWriter (`

```
OutputFormat format,
TextWriter writer,
Microsoft.ML.DataViewSchema schema )  [inline], [static]
```

Creates a writer of the request file format.

Writer is created for the specific file format and data view schema. The data view of the compatible schema can be written using the created writer. It is because some formats, e.g., CSV needs to know the schema in advance in order to write the header.

Parameters

format	File format.
writer	Underlying text writer.
schema	The data view schema.

Returns

A writer for the requested *format* .

Exceptions

NotSupportedException	

The documentation for this class was generated from the following file:

- DataView/DataViewWriterFactory.cs

2.6 IcsMonitor.Flows.FlowsDataViewSource< TInput, TRecord > Class Template Reference

An abstract class that also provides a method to create specific flow sources for different supported protocols.

Inherits FlowsDataViewSource.

Public Member Functions

- abstract Task< IDataView > LoadAndAggregateAsync< TKey > (MLContext mlContext, string inputCaptureFile, string inputLabelFile, TimeSpan windowTimeSpan, Func< FlowKey, TKey > getKey, CancellationToken cancellationToken)

  Loads and Aggregates ICS traffic from the given source.
- abstract IObservable< IDataView > ReadAndAggregateAsync< TKey > (MLContext mlContext, ICaptureDevice captureDevice, TimeSpan windowTimeSpan, Func< FlowKey, TKey > getKey, CancellationToken cancellationToken)

  Reads and Aggregates ICS traffic from the given source.
- abstract Task< IDataView > ReadAllAndAggregateAsync< TKey > (MLContext mlContext, ICaptureDevice captureDevice, TimeSpan windowTimeSpan, int windowCount, Func< FlowKey, TKey > getKey, Action< IEnumerable< object > > onNext, CancellationToken cancellationToken)

  Reads and Aggregates ICS traffic from the given source.
- abstract IDataView LoadFromCsvFile (MLContext mlContext, string file)

  Loads the data view from CSV source file.
- override Task< IDataView > **LoadAndAggregateAsync**< **TKey** > (MLContext mlContext, string inputCaptureFile, string inputLabelFile, TimeSpan windowTimeSpan, Func< FlowKey, TKey > getKey, CancellationToken cancellationToken)
- override IObservable< IDataView > **ReadAndAggregateAsync**< **TKey** > (MLContext mlContext, ICaptureDevice captureDevice, TimeSpan windowTimeSpan, Func< FlowKey, TKey > getKey, CancellationToken cancellationToken)
- override Task< IDataView > **ReadAllAndAggregateAsync**< **TKey** > (MLContext mlContext, ICaptureDevice captureDevice, TimeSpan windowTimeSpan, int windowCount, Func< FlowKey, TKey > getKey, Action< IEnumerable< object > > onNext, CancellationToken cancellationToken)
- abstract IObservable< TInput > LoadFromDevice (ICaptureDevice captureDevice, CancellationToken cancellationToken)

  Loads the input data from the capture devices and provides it in form of observable collection.
- abstract IObservable< TInput > LoadFromFile (string inputCaptureFile, string inputLabelFile, CancellationToken cancellationToken)

  Loads the input data from the input capture fileand provides it in form of observable collection.
- abstract IObservable< List< FlowRecord< TKey, TRecord > > > LoadDataFrom< TKey > (IObservable< TInput > source, TimeSpan windowSpan, Func< FlowKey, TKey > getKey)

  Loads data from the given source file and provides them in batches as observable sequence.
- abstract Task< IDataView > GetDataViewAsync< TKey > (MLContext ml, IObservable< FlowRecord< TKey, TRecord > > observable)

  Gets the dataview from the collection of records.
  This method implements the operation necessary to convert each record to the dataview row. As the record is a complex strcuture it is necessary to convert it to simple flat structure for which the dataview can be generated.

Static Public Member Functions

- static FlowsDataViewSource GetSource (IndustrialProtocol protocolType, IDictionary< string, string > configuration=null)

  Factory method that gets the particular flow source for the given protocolType .
- static IObservable< PacketRecord< Packet > > **LoadPacketsFromFile** (string inputCaptureFile, string inputLabelFile, CancellationToken cancellationToken)

  Loads packets and optionally labels from the input packet capture file and label file, respectively.

Parameters

inputCaptureFile	*the name of packet capture file.*
inputLabelFile	*the name of label file. If null then labels are not read.*
cancellationToken	*The cancellation token.*

Returns

An observable of packets loaded from the input file.

- static IObservable< PacketRecord< Packet > > **LoadPacketsFromDevice** (ICaptureDevice captureDevice, CancellationToken cancellationToken)

  Loads packets and optionally labels from the input packet capture device.

Parameters

inputCaptureFile	*the name of packet capture file.*
cancellationToken	*The cancellation token.*

Returns

An observable of packets loaded from the input file.

Protected Member Functions

- FlowsDataViewSource (IDictionary< string, string > configuration)

  Protected constructor for the data view source.
- FlowsDataViewSource (IDictionary< string, string > configuration)

  Creates a new instance of the class.

Protected Attributes

- Dictionary< string, string > **_configuration**

  The configuration collection.

Properties

- abstract IReadOnlyCollection< string > **FeatureColumns** `[get]`

  Collection of column names that are used to compute Features vector.
- Dictionary< string, string > **Configuration** `[get]`

  Gets the configuration as the key to value mapping.

2.6.1 Detailed Description

An abstract class that also provides a method to create specific flow sources for different supported protocols.

A typed version of FlowsDataViewSource that define other abstract methods.

Template Parameters

TRecord	The type of records to be provided by this data source.
TInput	The type of input object that this data source can process.

2.6.2 Constructor & Destructor Documentation

2.6.2.1 FlowsDataViewSource() [1/2]

```
IcsMonitor.Flows.FlowsDataViewSource< TInput, TRecord >.FlowsDataViewSource (
            IDictionary< string, string > configuration )  [inline], [protected]
```

Protected constructor for the data view source.

Parameters

configuration	The configuration of the data source.

2.6.2.2 FlowsDataViewSource() [2/2]

```
IcsMonitor.Flows.FlowsDataViewSource< TInput, TRecord >.FlowsDataViewSource (
            IDictionary< string, string > configuration )  [inline], [protected]
```

Creates a new instance of the class.

Parameters

configuration	

2.6.3 Member Function Documentation

2.6.3.1 GetDataViewAsync< TKey >()

```
abstract Task< IDataView > IcsMonitor.Flows.FlowsDataViewSource< TInput, TRecord >.GetDataViewAsync< TKey > (
            MLContext ml,
            IObservable< FlowRecord< TKey, TRecord > > observable )  [pure virtual]
```

Gets the dataview from the collection of records.

This method implements the operation necessary to convert each record to the dataview row. As the record is a complex strcuture it is necessary to convert it to simple flat structure for which the dataview can be generated.

Parameters

enumerable	An input enumerable of records to produce the data view.

Returns

A data view that represents the input observable.

2.6.3.2 GetSource()

```
static FlowsDataViewSource IcsMonitor.Flows.FlowsDataViewSource< TInput,
TRecord >.GetSource (
            IndustrialProtocol protocolType,
            IDictionary< string, string > configuration = null )  [inline], [static]
```

Factory method that gets the particular flow source for the given *protocolType* .

Parameters

protocolType	The type of the protocol.

Returns

A flow source object for the specific *protocolType* .

Exceptions

NotImplementedException	

2.6.3.3 LoadAndAggregateAsync< TKey >()

```
abstract Task< IDataView > IcsMonitor.Flows.FlowsDataViewSource<
TInput, TRecord >.LoadAndAggregateAsync< TKey > (
            MLContext mlContext,
            string inputCaptureFile,
            string inputLabelFile,
            TimeSpan windowTimeSpan,
            Func< FlowKey, TKey > getKey,
            CancellationToken cancellationToken )  [pure virtual]
```

Loads and Aggregates ICS traffic from the given source.

Template Parameters

TKey	The type of the flow key.

Parameters

mlContext	The ML context object required for some data view related operations.

Parameters

inputCaptureFile	The input capture file name.
inputLabelFile	the input label file name.
windowTimeSpan	The size of the time-aggregation window.
getKey	The function used to get key from the flow record.
cancellationToken	The cancellation token.

Returns

The task that when the method completes provide loaded data view.

2.6.3.4 LoadDataFrom< TKey >()

```
abstract IObservable< List< FlowRecord< TKey, TRecord > > >
IcsMonitor.Flows.FlowsDataViewSource< TInput, TRecord >.LoadDataFrom< TKey > (
            IObservable< TInput > source,
            TimeSpan windowSpan,
            Func< FlowKey, TKey > getKey )  [pure virtual]
```

Loads data from the given source file and provides them in batches as observable sequence.

Parameters

inputCaptureFile	An input capture file.
windowSpan	Size of window for collecting packets in the batches.
getKey	The aggregation key used to compose the flow records.

Returns

Observable collection of batches of records. Each batch represents a single window.

2.6.3.5 LoadFromCsvFile()

```
abstract IDataView IcsMonitor.Flows.FlowsDataViewSource< TInput,
TRecord >.LoadFromCsvFile (
            MLContext mlContext,
            string file )  [pure virtual]
```

Loads the data view from CSV source file.

Parameters

mlContext	The ML context object.
file	The CSV file name.

Returns

Implemented in IcsMonitor.Modbus.ModbusDataViewSource.

2.6.3.6 LoadFromDevice()

```
abstract IObservable< TInput > IcsMonitor.Flows.FlowsDataViewSource<
TInput, TRecord >.LoadFromDevice (
            ICaptureDevice captureDevice,
            CancellationToken cancellationToken )  [pure virtual]
```

Loads the input data from the capture devices and provides it in form of observable collection.

Parameters

captureDevice	The input capture device.
cancellationToken	The cancellation token.

Returns

The observable collection of *TInput* records.

Implemented in IcsMonitor.Modbus.ModbusDataViewSource.

2.6.3.7 LoadFromFile()

```
abstract IObservable< TInput > IcsMonitor.Flows.FlowsDataViewSource<
TInput, TRecord >.LoadFromFile (
            string inputCaptureFile,
            string inputLabelFile,
            CancellationToken cancellationToken )  [pure virtual]
```

Loads the input data from the input capture fileand provides it in form of observable collection.

Parameters

inputCaptureFile	The input capture file.
inputLabelFile	The input label file.
cancellationToken	The cancellaiton token.

Returns

Implemented in IcsMonitor.Modbus.ModbusDataViewSource.

2.6.3.8 ReadAllAndAggregateAsync< TKey >() abstract Task< IDataView > IcsMonitor.Flows.FlowsDataViewSource< TInput, TRecord >.ReadAllAndAggregateAsync< TKey > (

```
MLContext mlContext,
ICaptureDevice captureDevice,
TimeSpan windowTimeSpan,
int windowCount,
Func< FlowKey, TKey > getKey,
Action< IEnumerable< object > > onNext,
CancellationToken cancellationToken )  [pure virtual]
```

Reads and Aggregates ICS traffic from the given source.

Template Parameters

TKey	The type of the flow key.

Parameters

mlContext	The ML context object required for some data view related operations.
captureDevice	The input capture device use to read the traffic from.
windowTimeSpan	The size of the time-aggregation window.
windowCount	The total count of windows to process.
getKey	The function used to get key from the flow record.
onNext	On next callback for observe new objects. This can be used, e.g., for progress reporting and logging.
cancellationToken	The canclellation token.

Returns

2.6.3.9 ReadAndAggregateAsync< TKey >() abstract IObservable< IDataView > IcsMonitor.Flows.FlowsDataViewSou TInput, TRecord >.ReadAndAggregateAsync< TKey > (

```
MLContext mlContext,
ICaptureDevice captureDevice,
TimeSpan windowTimeSpan,
Func< FlowKey, TKey > getKey,
CancellationToken cancellationToken )  [pure virtual]
```

Reads and Aggregates ICS traffic from the given source.

Template Parameters

TKey	The type of the flow key.

Parameters

mlContext	The ML context object required for some data view related operations.
captureDevice	The input capture device use to read the traffic from.

Parameters

windowTimeSpan	The size of the time-aggregation window.
getKey	The function used to get key from the flow record.
cancellationToken	The cancellation token.

Returns

The observable object providing data views of read and aggregated flow records.

The documentation for this class was generated from the following file:

- Flows/FlowsDataViewSource.cs

2.7 IcsMonitor.AnomalyDetection.IAnomalyDetectionModel< TOutput > Interface Template Reference

Defines common interface for anomaly detection models.

Public Member Functions

- void SaveToFile (MLContext mlContext, string path)

  Stores the model in the file on the given path.
- IEnumerable< TOutput > Transform (MLContext mlContext, IDataView source)

  Transforms the source dataview using the anomaly detection model.
- ClusteringMetrics Evaluate (MLContext mlContext, IDataView testData)

  Evaluates the model using testData to produce ClusteringMetrics.

2.7.1 Detailed Description

Defines common interface for anomaly detection models.

Template Parameters

TOutput	The output type of the model.

2.7.2 Member Function Documentation

2.7.2.1 Evaluate() `ClusteringMetrics IcsMonitor.AnomalyDetection.IAnomalyDetectionModel< TOutput >.Evaluate (`

```
MLContext mlContext,
IDataView testData )
```

Evaluates the model using *testData* to produce ClusteringMetrics.

Parameters

mlContext	The ML.NET context.
testData	The test data used to evaluate the model.

Returns

The metrics of the computed model.

2.7.2.2 SaveToFile()

```
void IcsMonitor.AnomalyDetection.IAnomalyDetectionModel< TOutput >.SaveToFile (
            MLContext mlContext,
            string path )
```

Stores the model in the file on the given path.

Parameters

mlContext	The ML.NET context.
path	The path of output file.

2.7.2.3 Transform()

```
IEnumerable< TOutput > IcsMonitor.AnomalyDetection.IAnomalyDetectionModel<
TOutput >.Transform (
            MLContext mlContext,
            IDataView source )
```

Transforms the *source* dataview using the anomaly detection model.

Parameters

mlContext	The ML.NET context.
source	The source dataview which rows are to be evaluated.

Returns

The output for each input row.

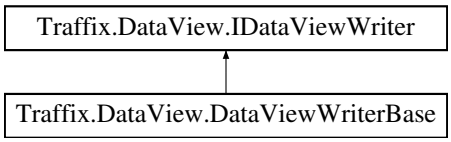
The documentation for this interface was generated from the following file:

- AnomalyDetection/IAnomalyDetectionModel.cs

2.8 Traffix.DataView.IDataViewWriter Interface Reference

A common interface for implementations of data view writers.

Inheritance diagram for Traffix.DataView.IDataViewWriter:



Public Member Functions

- void **BeginDocument** ()

  Writes the start of the document to the underlying text writer.
- void **EndDocument** ()

  Writes the end of the document to the underlying text writer.
- int AppendDataView (IDataView dataview)

  Writes the content of the data view using the underlying writer.

2.8.1 Detailed Description

A common interface for implementations of data view writers.

2.8.2 Member Function Documentation

2.8.2.1 AppendDataView() `int Traffix.DataView.IDataViewWriter.AppendDataView ( IDataView dataview )`

Writes the content of the data view using the underlying writer.

Parameters

dataview	

Implemented in Traffix.DataView.DataViewWriterBase.

The documentation for this interface was generated from the following file:

- DataView/IDataViewWriter.cs

2.9 IcsMonitor.Protocols.IecDataViewRecord Class Reference

This class represents the IEC record as used in ML's DataView.

Properties

- string **FlowLabel** [get, set]

  Flow identifier.
- string **Window** [get, set]

  Window label/identifier.
- DateTime **WindowStart** [get, set]

  Start of the window.
- TimeSpan **WindowDuration** [get, set]

  Duration of the window.
- string **FlowKey** [get]

  The flow key.
- int **Flows** = 1 [get, set]

  Number of IEC flows aggregated by the record.
- DateTime **StartDateTime** [get, set]

  Start time of the flow.
- string **SourceAddress** [get, set]

  Source address of the flow.
- string **DestinationAddress** [get, set]

  Destination address of the flow.
- int **SourcePort** [get, set]

  Source port of the flow.
- int **DestinationPort** [get, set]

  Destination address of the flow.
- int **Bytes** [get, set]

  Total number of bytes of the flow.
- int **Packets** [get, set]

  Number of packets of the flow.
- int **IecPacketLength** [get, set]

  Length of IEC packets.
- string **IecFrameFormat** [get, set]

  IEC frame format.
- string **AsduTypeIdentifier** [get, set]

  ASDU Type identifier of the IEC flow.
- int **AsduNumberOfItems** [get, set]

  ASDU number of items in IEC flow.
- string **CauseOfTransmission** [get, set]

  Cause of transmission value for the IEC flow.
- string **AsduOrg** [get, set]

  ASDU organization value.
- int **AsduAddress** [get, set]

  ASDU address value.
- string **OperationTag** [get, set]

  Computed operation tag.
- string[ ] **OperationTagVector** [get, set]

  A vector of existing operation tags of the IEC aggregated record.
- float[ ] **IecPacketLengthVector** [get, set]

  A vector of IEC packet lenghts of the current IEC aggregated record.
- float[ ] **AsduNumberOfItemsVector** [get, set]

  A vector of number of items of the current IEC aggregated record.

2.9.1 Detailed Description

This class represents the IEC record as used in ML's DataView.

The Data View record enables to combine individual IEC records in a single structure suitable for feature extraction in AD methods. Using this combination it is possible to count a number of different operations occured in the IEC conversation. For aggregated record, there is OperationTagVector that contains a vector of operation tags. To represents statistics for different operations
there are two counter vectors AsduNumberOfItemsVector and IecPacketLengthVector.

The documentation for this class was generated from the following file:

- Protocols/IEC/IecDataViewRecord.cs

2.10 IcsMonitor.Protocols.IecDataViewRecordFlowmon Class Reference

Represents IEC IPFIX record as defined by Flowmon. It is loaded by CsvHelper and thus its properties need to be annotated with

See also

CsvName

attribute.

Properties

- int **ExportCounter** `[get, set]`
  Export counter generated by Flowmon appliance.
- int **Bytes** `[get, set]`
  Number of bytes of the IEC flow.
- int **Packets** `[get, set]`
  Number of packet of the IEC flow.
- DateTime **StartDateTime** `[get, set]`
  the timestamp of the start of IEC flow.
- DateTime **EndDateTime** `[get, set]`
  The timestamp of the end of IEC flow.
- string **SourceAddress** `[get, set]`
  IEC flow source address.
- string **DestinationAddress** `[get, set]`
  IECflow destination address.
- string **SourcePort** `[get, set]`
  IEC flow source port.
- string **DestinationPort** `[get, set]`
  IEC flow destination port.
- string **IecPacketLength** `[get, set]`
  IEC packet length aggregated for IEC flow.
- string **IecFrameFormat** `[get, set]`
  IEC frame format.
- string **AsduTypeIdentifier** `[get, set]`

IEC ASDU type.

- string **AsduNumberOfItems** `[get, set]`

  A number of items in the ASDU message.
- string **CauseOfTransmission** `[get, set]`

  The cause of transmission value.
- string **AsduOrg** `[get, set]`

  ASDU organization value in IEC message.
- string **AsduAddress** `[get, set]`

  ASDU address as ocurred in IEC message.

2.10.1 Detailed Description

Represents IEC IPFIX record as defined by Flowmon. It is loaded by CsvHelper and thus its properties need to be annotated with

See also

CsvName

attribute.

The documentation for this class was generated from the following file:

- Protocols/IEC/IecDataViewRecord.cs

2.11 IcsMonitor.Protocols.IecDataViewRecordWireshark Class Reference

Represents IEC IPFIX record as produced by Wireshark IEC dissector.

Properties

- int **Bytes** `[get, set]`

  Size of the packet in bytes.
- int **Packets** = 1 `[get, set]`

  Number of packets representing the IEC packet/flow.
- DateTime **StartDateTime** `[get, set]`

  The timestamp of the packet.
- double **RelativeTime** `[get, set]`

  the relative time of the packet.
- string **SourceAddress** `[get, set]`

  The source address of the packet/flow.
- string **DestinationAddress** `[get, set]`

  The destination address pacekt/flow.
- string **SourcePort** `[get, set]`

  The source port of the packet/flow.
- string **DestinationPort** `[get, set]`

  The destination port of the packet/flow.
- string **IecPacketLength** `[get, set]`

The IEC packet length.

- string **IecFrameFormat** `[get, set]`

  The IEC packet format.
- string **AsduTypeIdentifier** `[get, set]`

  The ASDU type.
- string **AsduNumberOfItems** `[get, set]`

  A number of items in ASDU IEC packet.
- string **CauseOfTransmission** `[get, set]`

  The cause of transission of the IEC packet.
- string **AsduOrg** `[get, set]`

  Organization number of IEC packet.
- string **AsduAddress** `[get, set]`

  ASDU address of IEC packet.

2.11.1 Detailed Description

Represents IEC IPFIX record as produced by Wireshark IEC dissector.

The documentation for this class was generated from the following file:

- Protocols/IEC/IecDataViewRecord.cs

2.12 IcsMonitor.AnomalyDetection.TrafficProfileTrainer.InputFeatureData Class Reference

Reepresents input feature data.

Properties

- float[ ] **PreFeatures** `[get, set]`

  Features are represented as an array of float values.

2.12.1 Detailed Description

Reepresents input feature data.

The documentation for this class was generated from the following file:

- AnomalyDetection/TrafficProfileTrainer.cs

2.13 IcsMonitor.Flows.PacketAnnotationSourceFile.LabeledPackets Class Reference

A single packet label record. It contains annotation for reading and writing it direclty with CSVHelper library..

Properties

- int **PacketNumber** `[get, set]`
  Packet number column.
- int **PacketLabel** `[get, set]`
  Packet label column.

2.13.1 Detailed Description

A single packet label record. It contains annotation for reading and writing it direclty with CSVHelper library..

The documentation for this class was generated from the following file:

- Flows/PacketAnnotationSourceFile.cs

2.14 IcsMonitor.Modbus.ModbusCompact Class Reference

A compact version that only counts number of operations of each operation type.

Public Member Functions

- ModbusCompact (ref ModbusRawData data)
  Creates a new instance based in raw data .

Static Public Member Functions

- static ModbusCompact Aggregate (ModbusCompact x, ModbusCompact y)
  Aggregates two object into a new one.

Properties

- byte **UnitId** `[get]`
  Gets unit ID.
- int **ReadRequests** `[get]`
  Gets the number of all read requests.
- int **WriteRequests** `[get]`
  Gets the number of all write requests.
- int **DiagnosticRequests** `[get]`
  Gets the number of all diagnostic requests.
- int **OtherRequests** `[get]`
  Gets the number of other rquests.
- int **UndefinedRequests** `[get]`
  Gets the number of undefined requests.
- int **ResponsesSuccess** `[get]`
  Gets the number of successful responses.
- int **ResponsesError** `[get]`
  Gets the number of errorneous responses.
- int **MalformedRequests** `[get]`
  Gets the number of malforemd requests.
- int **MalformedResponses** `[get]`
  Get the number of malformed responses.

2.14.1 Detailed Description

A compact version that only counts number of operations of each operation type.

2.14.2 Constructor & Destructor Documentation

2.14.2.1 ModbusCompact() `IcsMonitor.Modbus.ModbusCompact.ModbusCompact (ref ModbusRawData data ) [inline]`

Creates a new instance based in raw *data* .

Parameters

data	Raw modbus data record.

2.14.3 Member Function Documentation

2.14.3.1 Aggregate() `static ModbusCompact IcsMonitor.Modbus.ModbusCompact.Aggregate (ModbusCompact x, ModbusCompact y ) [inline], [static]`

Aggregates two object into a new one.

Parameters

x	The first object.
y	The second object.

Returns

Aggregated modbus object.

The documentation for this class was generated from the following file:

- Protocols/MODBUS/ModbusCompact.cs

2.15 IcsMonitor.Modbus.ModbusDataViewRecord Class Reference

Represents a flattened record used as a typed version for corresponding Dataviews.

This class is computed from Flows.FlowRecord<ModbusCompact> and can be used for accesing dataview records.

Properties

- string **WindowLabel** `[get, set]`
  Window label. The flow can be collect in the window.
- DateTime WindowStart `[get, set]`
- TimeSpan **WindowDuration** `[get, set]`
  Duration of the window.
- string **FlowLabel** `[get, set]`
  The label of the flow. Can be used for classification.
- string **FlowKey** `[get, set]`
  The flow key. This field is required by the profile.
- float **ForwardMetricsDuration** `[get, set]`
  Duration of forward flow.
- float **ForwardMetricsOctets** `[get, set]`
  Number of octets in the forward flow.
- float **ForwardMetricsPackets** `[get, set]`
  Number of packets in the forward flow.
- long **ForwardMetricsFirstSeen** `[get, set]`
  Start time of the forward flow.
- long **ForwardMetricsLastSeen** `[get, set]`
  End time of the forward flow.
- float **ReverseMetricsDuration** `[get, set]`
  Duration of the reverse flow.
- float **ReverseMetricsOctets** `[get, set]`
  Number of octets in the reverse flow.
- float **ReverseMetricsPackets** `[get, set]`
  Number of packets in the reverse flow.
- long **ReverseMetricsFirstSeen** `[get, set]`
  Start time of the reverse flow.
- long **ReverseMetricsLastSeen** `[get, set]`
  End time of the reverse flow.
- float **DataUnitId** `[get, set]`
  The data unit IT value.
- float **DataReadRequests** `[get, set]`
  Number of read requests.
- float **DataWriteRequests** `[get, set]`
  Number of write requests.
- float **DataDiagnosticRequests** `[get, set]`
  Number of diagnostic requests.
- float **DataOtherRequests** `[get, set]`
  Number of other requests.
- float **DataUndefinedRequests** `[get, set]`
  Number of undefined requests.
- float **DataResponsesSuccess** `[get, set]`
  Number of correct responses.
- float **DataResponsesError** `[get, set]`
  Number of response with error code.
- float **DataMalformedRequests** `[get, set]`
  Number of malformed requests.
- float **DataMalformedResponses** `[get, set]`
  Number of malformed responses.

2.15.1 Detailed Description

Represents a flattened record used as a typed version for corresponding Dataviews.

This class is computed from Flows.FlowRecord<ModbusCompact> and can be used for accesing dataview records.

2.15.2 Property Documentation

2.15.2.1 WindowStart `DateTime IcsMonitor.Modbus.ModbusDataViewRecord.WindowStart [get], [set]`

Start of the window.

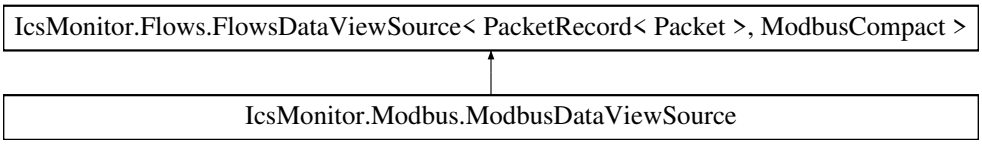
The documentation for this class was generated from the following file:

- Protocols/MODBUS/ModbusDataViewRecord.cs

2.16 IcsMonitor.Modbus.ModbusDataViewSource Class Reference

An implementation of data view source for MODBUS protocol.

Inheritance diagram for IcsMonitor.Modbus.ModbusDataViewSource:



Public Member Functions

- ModbusDataViewSource (IDictionary< string, string > configuration)

  Creates data view source.
- override IObservable< List< FlowRecord< TKey, ModbusCompact > > > **LoadDataFrom**< **TKey** > (IObservable< PacketRecord< Packet > > source, TimeSpan windowSpan, Func< FlowKey, TKey > getKey)
- override Task< IDataView > **GetDataViewAsync**< **TKey** > (MLContext ml, IObservable< FlowRecord< TKey, ModbusCompact > > observable)
- override IObservable< PacketRecord< Packet > > **LoadFromFile** (string inputCaptureFile, string inputLabelFile, CancellationToken cancellationToken)

  Loads the input data from the input capture fileand provides it in form of observable collection.

Parameters

inputCaptureFile	*The input capture file.*
inputLabelFile	*The input label file.*
cancellationToken	*The cancellaiton token.*

Returns

- override IObservable< PacketRecord< Packet > > **LoadFromDevice** (ICaptureDevice captureDevice, CancellationToken cancellationToken)

  Loads the input data from the capture devices and provides it in form of observable collection.

Parameters

captureDevice	*The input capture device.*
cancellationToken	*The cancellation token.*

Returns

The observable collection of TInput records.

- override IDataView **LoadFromCsvFile** (MLContext mlContext, string file)

  Loads the data view from CSV source file.

Parameters

mlContext	*The ML context object.*
file	*The CSV file name.*

Returns

- abstract Task< IDataView > LoadAndAggregateAsync< TKey > (MLContext mlContext, string input↩ CaptureFile, string inputLabelFile, TimeSpan windowTimeSpan, Func< FlowKey, TKey > getKey, CancellationToken cancellationToken)

  Loads and Aggregates ICS traffic from the given source.
- override Task< IDataView > **LoadAndAggregateAsync**< **TKey** > (MLContext mlContext, string inputCaptureFile, string inputLabelFile, TimeSpan windowTimeSpan, Func< FlowKey, TKey > getKey, CancellationToken cancellationToken)
- abstract IObservable< IDataView > ReadAndAggregateAsync< TKey > (MLContext mlContext, ICapture↩ Device captureDevice, TimeSpan windowTimeSpan, Func< FlowKey, TKey > getKey, CancellationToken cancellationToken)

  Reads and Aggregates ICS traffic from the given source.
- override IObservable< IDataView > **ReadAndAggregateAsync**< **TKey** > (MLContext mlContext, ICaptureDevice captureDevice, TimeSpan windowTimeSpan, Func< FlowKey, TKey > getKey, CancellationToken cancellationToken)
- abstract Task< IDataView > ReadAllAndAggregateAsync< TKey > (MLContext mlContext, ICaptureDevice captureDevice, TimeSpan windowTimeSpan, int windowCount, Func< FlowKey, TKey > getKey, Action< IEnumerable< object > > onNext, CancellationToken cancellationToken)

  Reads and Aggregates ICS traffic from the given source.
- override Task< IDataView > **ReadAllAndAggregateAsync**< **TKey** > (MLContext mlContext, ICapture↩ Device captureDevice, TimeSpan windowTimeSpan, int windowCount, Func< FlowKey, TKey > getKey, Action< IEnumerable< object > > onNext, CancellationToken cancellationToken)
- abstract IObservable< List< FlowRecord< TKey, TRecord > > > LoadDataFrom< TKey > (IObservable< TInput > source, TimeSpan windowSpan, Func< FlowKey, TKey > getKey)

  Loads data from the given source file and provides them in batches as observable sequence.
- abstract Task< IDataView > GetDataViewAsync< TKey > (MLContext ml, IObservable< FlowRecord< TKey, TRecord > > observable)

  Gets the dataview from the collection of records.
  This method implements the operation necessary to convert each record to the dataview row. As the record is a complex strcuture it is necessary to convert it to simple flat structure for which the dataview can be generated.

Static Public Member Functions

- static FlowsDataViewSource GetSource (IndustrialProtocol protocolType, IDictionary< string, string > configuration=null)

  Factory method that gets the particular flow source for the given protocolType .
- static IObservable< PacketRecord< Packet > > **LoadPacketsFromFile** (string inputCaptureFile, string inputLabelFile, CancellationToken cancellationToken)

  Loads packets and optionally labels from the input packet capture file and label file, respectively.

Parameters

inputCaptureFile	*the name of packet capture file.*
inputLabelFile	*the name of label file. If null then labels are not read.*
cancellationToken	*The cancellation token.*

Returns

An observable of packets loaded from the input file.

- static IObservable< PacketRecord< Packet > > **LoadPacketsFromDevice** (ICaptureDevice captureDevice, CancellationToken cancellationToken)

  Loads packets and optionally labels from the input packet capture device.

Parameters

inputCaptureFile	*the name of packet capture file.*
cancellationToken	*The cancellation token.*

Returns

An observable of packets loaded from the input file.

Protected Attributes

- Dictionary< string, string > **_configuration**

  The configuration collection.

Properties

- override IReadOnlyCollection< string > **FeatureColumns** `[get]`
- Dictionary< string, string > **Configuration** `[get]`

  Gets the configuration as the key to value mapping.

2.16.1 Detailed Description

An implementation of data view source for MODBUS protocol.

2.16.2 Constructor & Destructor Documentation

2.16.2.1 ModbusDataViewSource() `IcsMonitor.Modbus.ModbusDataViewSource.ModbusDataViewSource ( IDictionary< string, string > configuration ) [inline]`

Creates data view source.

Parameters

configuration	The configuration object.

2.16.3 Member Function Documentation

2.16.3.1 GetDataViewAsync< TKey >()

```
abstract Task< IDataView > IcsMonitor.Flows.FlowsDataViewSource<
TInput, TRecord >.GetDataViewAsync< TKey > (
            MLContext ml,
            IObservable< FlowRecord< TKey, TRecord > > observable )  [pure virtual], [inherited]
```

Gets the dataview from the collection of records.

This method implements the operation necessary to convert each record to the dataview row. As the record is a complex strcuture it is necessary to convert it to simple flat structure for which the dataview can be generated.

Parameters

enumerable	An input enumerable of records to produce the data view.

Returns

A data view that represents the input observable.

2.16.3.2 GetSource()

```
static FlowsDataViewSource IcsMonitor.Flows.FlowsDataViewSource< TInput,
TRecord >.GetSource (
            IndustrialProtocol protocolType,
            IDictionary< string, string > configuration = null )  [inline], [static], [inherited]
```

Factory method that gets the particular flow source for the given *protocolType* .

Parameters

protocolType	The type of the protocol.

Returns

A flow source object for the specific *protocolType* .

Exceptions

NotImplementedException	

2.16.3.3 LoadAndAggregateAsync< TKey >() abstract Task< IDataView > IcsMonitor.Flows.FlowsDataViewSource< TInput, TRecord >.LoadAndAggregateAsync< TKey > (

```
            MLContext mlContext,
            string inputCaptureFile,
            string inputLabelFile,
            TimeSpan windowTimeSpan,
            Func< FlowKey, TKey > getKey,
            CancellationToken cancellationToken )  [pure virtual], [inherited]
```

Loads and Aggregates ICS traffic from the given source.

Template Parameters

TKey	The type of the flow key.

Parameters

mlContext	The ML context object required for some data view related operations.
inputCaptureFile	The input capture file name.
inputLabelFile	the input label file name.
windowTimeSpan	The size of the time-aggregation window.
getKey	The function used to get key from the flow record.
cancellationToken	The cancellation token.

Returns

The task that when the method completes provide loaded data view.

2.16.3.4 LoadDataFrom< TKey >() abstract IObservable< List< FlowRecord< TKey, TRecord > > > IcsMonitor.Flows.FlowsDataViewSource< TInput, TRecord >.LoadDataFrom< TKey > (

```
            IObservable< TInput > source,
            TimeSpan windowSpan,
            Func< FlowKey, TKey > getKey )  [pure virtual], [inherited]
```

Loads data from the given source file and provides them in batches as observable sequence.

Parameters

inputCaptureFile	An input capture file.
windowSpan	Size of window for collecting packets in the batches.
getKey	The aggregation key used to compose the flow records.

Returns

Observable collection of batches of records. Each batch represents a single window.

2.16.3.5 ReadAllAndAggregateAsync< TKey >() abstract Task< IDataView > IcsMonitor.Flows.FlowsDataViewSource<

```
TInput, TRecord >.ReadAllAndAggregateAsync< TKey > (
            MLContext mlContext,
            ICaptureDevice captureDevice,
            TimeSpan windowTimeSpan,
            int windowCount,
            Func< FlowKey, TKey > getKey,
            Action< IEnumerable< object > > onNext,
            CancellationToken cancellationToken )  [pure virtual], [inherited]
```

Reads and Aggregates ICS traffic from the given source.

Template Parameters

TKey	The type of the flow key.

Parameters

mlContext	The ML context object required for some data view related operations.
captureDevice	The input capture device use to read the traffic from.
windowTimeSpan	The size of the time-aggregation window.
windowCount	The total count of windows to process.
getKey	The function used to get key from the flow record.
onNext	On next callback for observe new objects. This can be used, e.g., for progress reporting and logging.
cancellationToken	The canclellation token.

Returns

2.16.3.6 ReadAndAggregateAsync< TKey >() abstract IObservable< IDataView > IcsMonitor.Flows.FlowsDataViewSo

```
TInput, TRecord >.ReadAndAggregateAsync< TKey > (
            MLContext mlContext,
            ICaptureDevice captureDevice,
            TimeSpan windowTimeSpan,
            Func< FlowKey, TKey > getKey,
            CancellationToken cancellationToken )  [pure virtual], [inherited]
```

Reads and Aggregates ICS traffic from the given source.

Template Parameters

TKey	The type of the flow key.

Parameters

mlContext	The ML context object required for some data view related operations.
captureDevice	The input capture device use to read the traffic from.

Parameters

windowTimeSpan	The size of the time-aggregation window.
getKey	The function used to get key from the flow record.
cancellationToken	The cancellation token.

Returns

The observable object providing data views of read and aggregated flow records.

The documentation for this class was generated from the following file:

- Protocols/MODBUS/ModbusDataViewSource.cs

2.17 IcsMonitor.Modbus.ModbusFlowProcessor< TKey > Class Template Reference

Flow processor for extracting MODBUS related information from bidirectional flows.

Inherits FlowProcessor< PacketRecord< Packet >, FlowKey, FlowRecord< TKey, ModbusCompact > >.

Public Member Functions

- ModbusFlowProcessor (string label, DateTime start, TimeSpan duration, Func< FlowKey, TKey > getKey)

  Creates the flow processor of the given index.

Protected Member Functions

- override FlowRecord< TKey, ModbusCompact > **Aggregate** (FlowRecord< TKey, ModbusCompact > arg1, FlowRecord< TKey, ModbusCompact > arg2)
- override void **Update** (FlowRecord< TKey, ModbusCompact > record, PacketRecord< Packet > packet)
- override FlowRecord< TKey, ModbusCompact > **Create** (PacketRecord< Packet > arg)
- override FlowKey GetFlowKey (PacketRecord< Packet > source)

  Gets the flow key from the given source packet.

Properties

- string **Label** `[get]`

  A label assigned to the processor. Can be used to uniquely identify the processor among a list of processors..
- DateTime **Start** `[get]`

  Defines the timestamp for the first samples processes by the processor.
- TimeSpan **Duration** `[get]`

  Defines the duration/interval for which the processor accepts the samples.

2.17.1 Detailed Description

Flow processor for extracting MODBUS related information from bidirectional flows.

2.17.2 Constructor & Destructor Documentation

2.17.2.1 ModbusFlowProcessor() IcsMonitor.Modbus.ModbusFlowProcessor< TKey >.ModbusFlowProcessor (

```
            string label,
            DateTime start,
            TimeSpan duration,
            Func< FlowKey, TKey > getKey )  [inline]
```

Creates the flow processor of the given index.

Parameters

index	The identification of the flow processor.

2.17.3 Member Function Documentation

2.17.3.1 GetFlowKey() override FlowKey IcsMonitor.Modbus.ModbusFlowProcessor< TKey >.GetFlowKey (

```
            PacketRecord< Packet > source )  [inline], [protected]
```

Gets the flow key from the given *source* packet.

Parameters

source	the source packet object.

Returns

FlowKey extracted from the *source* packet.

The documentation for this class was generated from the following file:

- Protocols/MODBUS/ModbusFlowProcessor.cs

2.18 IcsMonitor.Modbus.ModbusRawData Struct Reference

A full version of the MODBUS flow record.

2.18.1 Detailed Description

A full version of the MODBUS flow record.

The documentation for this struct was generated from the following file:

- Protocols/MODBUS/ModbusRawData.cs

2.19 IcsMonitor.AnomalyDetection.ModelTrainer Class Reference

Represents anomaly detection model trainer.

It provides methods for training different anomaly detection methods.

Public Member Functions

- **ModelTrainer** (MLContext mlContext)

  Creates a new trainer in the given context.
- ClusterModel TrainKMeansAnomalyDetector (IDataView trainingDataView, ClusterModel.Options options, string featuresColumnName, params string[ ] slotNames)

  Creates a single K-Means model of network traffic for the given input and configuration.

2.19.1 Detailed Description

Represents anomaly detection model trainer.

It provides methods for training different anomaly detection methods.

2.19.2 Member Function Documentation

2.19.2.1 TrainKMeansAnomalyDetector()

```
ClusterModel IcsMonitor.AnomalyDetection.ModelTrainer.↩
TrainKMeansAnomalyDetector (
            IDataView trainingDataView,
            ClusterModel.Options options,
            string featuresColumnName,
            params string[] slotNames )  [inline]
```

Creates a single K-Means model of network traffic for the given input and configuration.

Parameters

trainingDataView	The training data.
options	The method options.
featuresColumnName	A name of the features column.
slotNames	An array of slot names of the features vector.

Returns

New ClusterModel model created from the given training data.

The documentation for this class was generated from the following file:

- AnomalyDetection/ModelTrainer.cs

2.20 IcsMonitor.Utils.OptionHelper Class Reference

A helper class for processing command line options.

Public Member Functions

- delegate bool TryParse< T > (string input, out T value)

  Defines type for delegates used for safe option parsing.

Static Public Member Functions

- static bool TryGetValueOrDefault (this CommandOption option, string defaultValue, out string value)

  Gets an option value or a provided default value.
- static bool TryGetValueOrError (this CommandOption option, Action OnValueMissingError, out string value)

  Gets an option value or executes the specified action if value there is not any value.
- static bool TryParseValueOrDefault< TValue > (this CommandOption option, TryParse< TValue > tryParse, TValue defaultValue, Action< string > OnError, out TValue value)

  Gets an option value or a provided default value.
- static bool TryParseValueOrError< TValue > (this CommandOption option, TryParse< TValue > tryParse, Action OnValueMissingError, Action< string > OnParseError, out TValue value)

  Gets an option value or a execute the given action on error.

2.20.1 Detailed Description

A helper class for processing command line options.

2.20.2 Member Function Documentation

2.20.2.1 TryGetValueOrDefault() `static bool IcsMonitor.Utils.OptionHelper.TryGetValueOrDefault (`

```
            this CommandOption option,
            string defaultValue,
            out string value )  [inline], [static]
```

Gets an option value or a provided default value.

Parameters

option	The command option object.
defaultValue	The defalt value used if option does not have any value.
value	The output value.

Returns

true if option's value was used.

2.20.2.2 TryGetValueOrError() `static bool IcsMonitor.Utils.OptionHelper.TryGetValueOrError (`

```
this CommandOption option,
Action OnValueMissingError,
out string value )  [inline], [static]
```

Gets an option value or executes the specified action if value there is not any value.

Parameters

option	The command option object.
OnValueMissingError	The action to be executed if not value can be used.
value	The output value

Returns

true if option's value was used.

2.20.2.3 TryParse< T >() `delegate bool IcsMonitor.Utils.OptionHelper.TryParse< T > (`

```
string input,
out T value )
```

Defines type for delegates used for safe option parsing.

Template Parameters

T	The type of the option.

Parameters

input	The input string.
value	<the parsed value./param> **Returns** true if option was parsed. false for any erro during parsing.

2.20.2.4 TryParseValueOrDefault< TValue >() `static bool IcsMonitor.Utils.OptionHelper.TryParse↩ ValueOrDefault< TValue > (`

```
this CommandOption option,
```

```
            TryParse< TValue > tryParse,
            TValue defaultValue,
            Action< string > OnError,
            out TValue value )  [inline], [static]
```

Gets an option value or a provided default value.

Template Parameters

TValue	The type of the output value.

Parameters

option	The command option object.
tryParse	The method used to parse the option.
defaultValue	The default value to use.
OnError	The action executed when error ocurred during parsing.
value	The output value.

Returns

true if option's value was used.

2.20.2.5 TryParseValueOrError< TValue >() `static bool IcsMonitor.Utils.OptionHelper.TryParseValueOrError< TValue > (`

```
            this CommandOption option,
            TryParse< TValue > tryParse,
            Action OnValueMissingError,
            Action< string > OnParseError,
            out TValue value )  [inline], [static]
```

Gets an option value or a execute the given action on error.

Template Parameters

TValue	The type of the output value.

Parameters

option	The command option object.
tryParse	The method used to parse the option.
OnValueMissingError	The action executed when option value is missing.
OnParseError	The action executed when error ocurred during parsing.
value	>The output value.

Returns

true if option's value was used.

The documentation for this class was generated from the following file:

- Utils/OptionHelper.cs

2.21 IcsMonitor.AnomalyDetection.ClusterModel.Options Class Reference

Defines the options for creating the model.

Properties

- int **NumberOfClusters** `[get, set]`
  The exact number of clusters.

2.21.1 Detailed Description

Defines the options for creating the model.

The documentation for this class was generated from the following file:

- AnomalyDetection/ClusterModel.cs

2.22 IcsMonitor.AnomalyDetection.ClusterModel.Output Class Reference

Represents the cluster prediction data type. This is the output type from the prediction.

See tutorial on K-Means clustering for more details: https://docs.microsoft.com/en-us/dotnet/machine-lear

Public Attributes

- string **WindowLabel**
  Gets the window label.
- DateTime **WindowStart**
  Gets the timestamp of the start of the window.
- TimeSpan **WindowDuration**
  Gets the window duration.
- string **FlowLabel**
  The label of the flow.
- string **FlowKey**
  The key of the putput record.

Properties

- uint **ClusterId** `[get, set]`

  Contains the ID of the predicted cluster.
  The underlying algorithm requires that predicted cluster id column has name 'PredictedLabel'.
- float **Distance** `[get]`

  Gets the distance to the predicted cluster centroid.
- float[ ] **Distances** `[get, set]`

  Contains an array with squared Euclidean distances to the cluster centroids. The array length is equal to the number of clusters.
  The underlying algorithm requires that predicted cluster id column has name 'Score'.
- float **Variance** `[get, set]`

  The computed variance for the predicted cluster.
  Compare this value to the distance for decision of whether to accept the point or not.
- float[ ] **Features** `[get, set]`

  Collection of features used as an input for the prediction algorithm.
  The underlying algorithm requires that this column has name 'Features'.

2.22.1 Detailed Description

Represents the cluster prediction data type. This is the output type from the prediction.

See tutorial on K-Means clustering for more details: https://docs.microsoft.com/en-us/dotnet/machine-lear

The documentation for this class was generated from the following file:

- AnomalyDetection/ClusterModel.cs

2.23 IcsMonitor.AnomalyDetection.TrafficProfileTrainer.OutputFeatureData Class Reference

Represents output feature data.

Properties

- float[ ] **Features** `[get, set]`

  Features are represented as an array of float values.

2.23.1 Detailed Description

Represents output feature data.

The documentation for this class was generated from the following file:

- AnomalyDetection/TrafficProfileTrainer.cs

2.24 IcsMonitor.Flows.PacketAnnotationSourceFile Class Reference

Represents a packet annotation source file.

Packet annotation is a CSV file that matches labesl to packet numbers.

Classes

- class LabeledPackets

  A single packet label record. It contains annotation for reading and writing it direclty with CSVHelper library..

Static Public Member Functions

- static IEnumerable< LabeledPackets > ReadLabels (string csvPath)

  Reads the labels from the CSV file and provides them as enumerable.
- static IObservable< LabeledPackets > ReadLabelsAsync (string csvPath)

  Reads the packet labels from the CSV file and provides them as observable.

2.24.1 Detailed Description

Represents a packet annotation source file.

Packet annotation is a CSV file that matches labesl to packet numbers.

2.24.2 Member Function Documentation

2.24.2.1 ReadLabels() `static IEnumerable< LabeledPackets > IcsMonitor.Flows.PacketAnnotation↩SourceFile.ReadLabels (`

`string csvPath ) [inline], [static]`

Reads the labels from the CSV file and provides them as enumerable.

Parameters

csvPath	The source CSV file.

Returns

The enumerable of packet label records.

2.24.2.2 ReadLabelsAsync() `static IObservable< LabeledPackets > IcsMonitor.Flows.Packet↩AnnotationSourceFile.ReadLabelsAsync (`

`string csvPath ) [inline], [static]`

Reads the packet labels from the CSV file and provides them as observable.

Parameters

csvPath	The source CSV file.

Returns

The observable of packet label records.

The documentation for this class was generated from the following file:

- Flows/PacketAnnotationSourceFile.cs

2.25 IcsMonitor.Flows.PacketDeviceSource Class Reference

Supports observable for the capture device.

Inherits IDisposable.

Public Member Functions

- void **Dispose** ()
- void **Close** ()

  Closes the packet source provider.

Static Public Member Functions

- static IDisposable Subscribe (ICaptureDevice captureDevice, IObserver< PacketRecord< Packet > > observer, CancellationToken cancellationToken)

  Subsrcibes the observer to the newly created packet source provider based on captureDevice .

Properties

- int **PacketCount** `[get]`

  Gets the number of packets provided so far.

2.25.1 Detailed Description

Supports observable for the capture device.

2.25.2 Member Function Documentation

2.25.2.1 Subscribe()

```
static IDisposable IcsMonitor.Flows.PacketDeviceSource.Subscribe (
    ICaptureDevice captureDevice,
    IObserver< PacketRecord< Packet > > observer,
    CancellationToken cancellationToken ) [inline], [static]
```

Subsrcibes the *observer* to the newly created packet source provider based on *captureDevice* .

Parameters

captureDevice	The capture device.
observer	the observer object.
cancellationToken	The cancellation token.

Returns

Disposable that can be used to unsubscribe from the observable.

The documentation for this class was generated from the following file:

- Flows/FlowsDataViewSource.cs

2.26 IcsMonitor.AnomalyDetection.TrafficProfile Class Reference

Represents traffic profile that consists of a collection of models. The profile is used for anomaly detection provided the network traffic.

Public Member Functions

- FlowsDataViewSource GetSource ()

  Gets the data view source object for the protocol type of the current profile.
  Each protocol type has a different source object. The factory object is FlowsDataViewSource that can provide data view source instance for all supported ICS protocols.
- IEnumerable< FlowScore > Predict (IDataView testData)

  Performs analysis of the input data and generates an enumerable with predicted/classified output. The given testData are first preprocessed using InputTransformer and then all models are applied. FlowScore object is generated for each input record.
- IDataView Transform (IDataView testData)

  Transform the input data view and produces a set of FlowScore represented as IDataView.
- void SaveToFile (string path)

  Stores the profile to the file.

Static Public Member Functions

- static TrafficProfile LoadFromFile (MLContext mlContext, string path)

  Loads the profile from the given file.

Properties

- IDictionary< string, string > **Configuration** `[get]`

  The configuration map. It is a collection of key-value pairs.
- IndustrialProtocol **ProtocolType** `[get]`

  The protocol name for which this profile was created.
- TimeSpan **WindowTimeSpan** `[get]`

  The size of time window used for creating the profile.
- ITransformer **InputTransformer** `[get]`

  Gets the input data transformer. It takes input data and performs several transformation to prepare them for evaluation by models.
- ClusterModel[ ] **Models** `[get]`

  Gets models of the current profile.
- DataViewSchema **InputSchema** `[get]`

  Gets the schema required for the input dataview.

2.26.1 Detailed Description

Represents traffic profile that consists of a collection of models. The profile is used for anomaly detection provided the network traffic.

2.26.2 Member Function Documentation

2.26.2.1 GetSource() `FlowsDataViewSource IcsMonitor.AnomalyDetection.TrafficProfile.GetSource ( ) [inline]`

Gets the data view source object for the protocol type of the current profile.

Each protocol type has a different source object. The factory object is FlowsDataViewSource that can provide data view source instance for all supported ICS protocols.

Returns

The flows datav view source object usable with the current profile.

2.26.2.2 LoadFromFile() `static TrafficProfile IcsMonitor.AnomalyDetection.TrafficProfile.LoadFromFile ( MLContext mlContext, string path ) [inline], [static]`

Loads the profile from the given file.

Parameters

mlContext	The ML.NET context.
path	Path to the profile file.

Returns

Profile loaded from the specifed file.

2.26.2.3 Predict() `IEnumerable< FlowScore > IcsMonitor.AnomalyDetection.TrafficProfile.Predict ( IDataView testData ) [inline]`

Performs analysis of the input data and generates an enumerable with predicted/classified output. The given *testData* are first preprocessed using InputTransformer and then all models are applied. FlowScore object is generated for each input record.

Parameters

testData	The input test data.

Returns

A collection of FlowScore objects.

2.26.2.4 SaveToFile() `void IcsMonitor.AnomalyDetection.TrafficProfile.SaveToFile ( string path ) [inline]`

Stores the profile to the file.

Parameters

path	Path to file to store the profile.

2.26.2.5 Transform() `IDataView IcsMonitor.AnomalyDetection.TrafficProfile.Transform ( IDataView testData ) [inline]`

Transform the input data view and produces a set of FlowScore represented as IDataView.

Parameters

testData	

Returns

A dataview consisting of the results of application of the profile to *testData* .

The documentation for this class was generated from the following file:

- AnomalyDetection/TrafficProfile.cs

2.27 IcsMonitor.AnomalyDetection.TrafficProfileTrainer Class Reference

Trainer for creating a profile based on the provided dataview.

Classes

- class InputFeatureData

  Reepresents input feature data.
- class OutputFeatureData

  Represents output feature data.

Public Member Functions

- TrafficProfileTrainer (MLContext ml, int pcaRank, IndustrialProtocol protocolType, string[ ] featureColumns, TimeSpan windowTimeSpan, string[ ] tags=null)

  Creates a new instance of the trainer.
- ITransformer GetTransformer (IDataView dataview, Func< IEstimator< ITransformer >, IEstimator< ITransformer > > featureTransformer)

  Gets the input data transformer fitted to the provided Dataview.
  The input data transformer creates features vector based on the fields as specified for the protocol, normalizes the input data using min-max method and reduces the data dimensions using PCA method. This transformation can be used to prepare data for the profile trainer.
- Func< IEstimator< ITransformer >, IEstimator< ITransformer > > PcaTransformer (int rank)

  Uses PCA method to compute features from pre-features.
- IEstimator< ITransformer > DirectTransformer (IEstimator< ITransformer > estimator)

  The direct transformation. It just uses the input features as output features.
- IEstimator< ITransformer > AverageTransformer (IEstimator< ITransformer > estimator)

  Computes MIN,MAX,AVG,STDEV from the prefeatures.
- TrafficProfile Fit (string profileName, Dictionary< string, string > configuration, Func< IEstimator< ITransformer >, IEstimator< ITransformer > > featureTransformer, int[ ] clusterCountVector, int maxModelCount, IDataView dataview)

  Creates a profile for the source dataview .
  The profile consists of maxModelCount models which are selected form models computed for clusters in range between minClusters and maxClusters .

2.27.1 Detailed Description

Trainer for creating a profile based on the provided dataview.

2.27.2 Constructor & Destructor Documentation

2.27.2.1 TrafficProfileTrainer() `IcsMonitor.AnomalyDetection.TrafficProfileTrainer.TrafficProfileTrainer (`

```
            MLContext ml,
            int pcaRank,
            IndustrialProtocol protocolType,
            string[] featureColumns,
            TimeSpan windowTimeSpan,
            string[] tags = null )  [inline]
```

Creates a new instance of the trainer.

Parameters

ml	The ML.NET context.
pcaRank	The rank of the PCA space. If set to 0, then PCA is not used.
protocolType	The target protocol type.
featureColumns	Defines columns that contains features used in the model.
windowTimeSpan	The size of time window used for aggregating the input data.
tags	For some protocol, the model contains dynamic tags.

2.27.3 Member Function Documentation

2.27.3.1 AverageTransformer()

```
IEstimator< ITransformer > IcsMonitor.AnomalyDetection.TrafficProfileTrainer.AverageTransformer (
            IEstimator< ITransformer > estimator )  [inline]
```

Computes MIN,MAX,AVG,STDEV from the prefeatures.

Parameters

estimator	The input estimator.

Returns

The output estimator with transformer applied.

2.27.3.2 DirectTransformer()

```
IEstimator< ITransformer > IcsMonitor.AnomalyDetection.TrafficProfileTrainer.DirectTransformer (
            IEstimator< ITransformer > estimator )  [inline]
```

The direct transformation. It just uses the input features as output features.

Parameters

estimator	The input estimator.

Returns

A new estimator with a direct transformer applied.

2.27.3.3 Fit()

```
TrafficProfile IcsMonitor.AnomalyDetection.TrafficProfileTrainer.Fit (
            string profileName,
            Dictionary< string, string > configuration,
            Func< IEstimator< ITransformer >, IEstimator< ITransformer > > featureTransformer,
            int[] clusterCountVector,
            int maxModelCount,
            IDataView dataview )  [inline]
```

Creates a profile for the source *dataview* .

The profile consists of *maxModelCount* models which are selected form models computed for clusters in range between *minClusters* and *maxClusters* .

Parameters

profileName	The profile name.
dataview	An input data view with training data.
clusterCountVector	An array of cluster count values.
maxModelCount	The required number of models in the profile.

Returns

The profile for the traffic.

2.27.3.4 GetTransformer()

```
ITransformer IcsMonitor.AnomalyDetection.TrafficProfileTrainer.GetTransformer (
            IDataView dataview,
            Func< IEstimator< ITransformer >, IEstimator< ITransformer > > featureTransformer
) [inline]
```

Gets the input data transformer fitted to the provided Dataview.

The input data transformer creates features vector based on the fields as specified for the protocol, normalizes the input data using min-max method and reduces the data dimensions using PCA method. This transformation can be used to prepare data for the profile trainer.

Parameters

dataview	The data view used to fit the input data transformer.
featureTransformer	The transformer used to compute actual input features from the candidate features.

Returns

The transformer for input data transformation fitted to the provided Dataview.

2.27.3.5 PcaTransformer()

```
Func< IEstimator< ITransformer >, IEstimator< ITransformer > >
IcsMonitor.AnomalyDetection.TrafficProfileTrainer.PcaTransformer (
            int rank )
```

Uses PCA method to compute features from pre-features.

Parameters

rank	The rank of the resulting PCA.

Returns

The input to output estimator function.

The documentation for this class was generated from the following file:

- AnomalyDetection/TrafficProfileTrainer.cs

2.28 Traffix.DataView.TraffixTransformsCatalog Class Reference

The extension class implementing project's specific transformers.

Static Public Member Functions

- static EstimatorChain< ColumnConcatenatingTransformer > CreateFeatureVector (this TransformsCatalog transforms, DataViewSchema schema, string featureColumnName, params string[ ] sourceColumns)

  Provides a transformer that creates a feature vector from the given columns. It converts all values in the source columns to floating-point numbers before creating the feature vector column.

2.28.1 Detailed Description

The extension class implementing project's specific transformers.

2.28.2 Member Function Documentation

2.28.2.1 CreateFeatureVector() `static EstimatorChain< ColumnConcatenatingTransformer > Traffix.DataView.TraffixTransformsCatalog.CreateFeatureVector ( this TransformsCatalog transforms, DataViewSchema schema, string featureColumnName, params string[] sourceColumns ) [inline], [static]`

Provides a transformer that creates a feature vector from the given columns. It converts all values in the source columns to floating-point numbers before creating the feature vector column.

Parameters

featureColumnName	The name of the resulting feature vector column.
sourceColumns	The array of source columns.

Returns

The transformer that can be a part of the ML pipeline.

The documentation for this class was generated from the following file:

- DataView/DataViewTransformsCatalog.cs

2.29 IcsMonitor.Utils.ZipEntryYamlIO Class Reference

An extension class for I/O operations with ZipArchiveEntry.

Static Public Member Functions

- static void WriteYaml< T > (this ZipArchiveEntry entry, T value)

  Writes value as yaml file to Zip archive entry .
- static T ReadYaml< T > (this ZipArchiveEntry entry)

  Reads the Zip archive entry as YAML document of type T .

2.29.1 Detailed Description

An extension class for I/O operations with ZipArchiveEntry.

2.29.2 Member Function Documentation

2.29.2.1 ReadYaml< T >() `static T IcsMonitor.Utils.ZipEntryYamlIO.ReadYaml< T > ( this ZipArchiveEntry entry ) [inline], [static]`

Reads the Zip archive *entry* as YAML document of type *T* .

Template Parameters

T	The type of the document.

Parameters

entry	The Zip archive entry.

Returns

Object of type *T* read from the given *entry* .

2.29.2.2 WriteYaml< T >() `static void IcsMonitor.Utils.ZipEntryYamlIO.WriteYaml< T > ( this ZipArchiveEntry entry, T value ) [inline], [static]`

Writes *value* as yaml file to Zip archive *entry* .

Template Parameters

T	The type of object to write.

Parameters

entry	Zip archive entry.
value	Value of the object to write.

The documentation for this class was generated from the following file:

- Utils/ZipEntryYamlIO.cs

Index