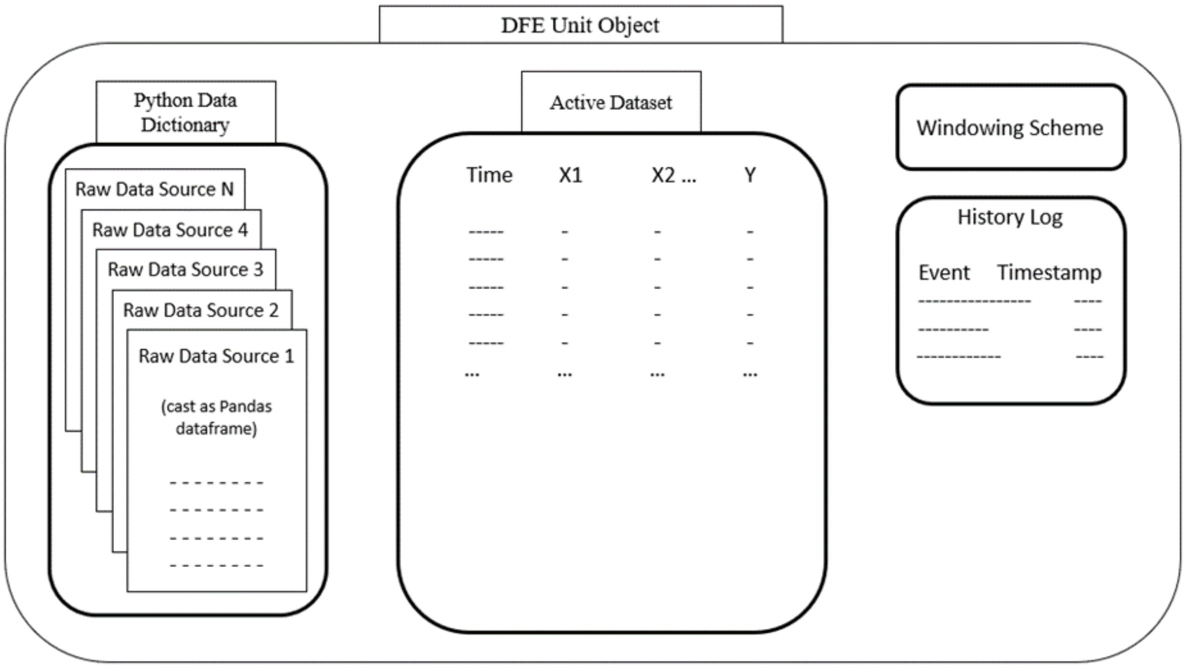
Data Fusion Explorer Module Documentation

Devon Martin

Last Updated: 2023-05-12

Object Structure

The DFE Object contains a list of inputted data sources, an active dataset where fusion occurs, and miscellaneous metadata such as a windowing scheme and log of commands inputted.



To get started, you will have to create a DFE object, which can then be called with various functions.

```
>>> dfeo = DFE_object()
```

Callable Functions

I. Data Uploading

import_from_pandas(df, **kwargs)

imports data from a provided Pandas DataFrame (df). Optional keyword arguments for metadata like dataset name (dataset_name, as string), time column (time_col, as integer), output column (y, as integer), and categorical columns (categorical, as integer or list of integers). If a name is not supplied, the ith uploaded dataset will be called “Entry_i”.

drop_dataset(dataset_name)

removes the dataset with name dataset_name from the DFE object. To find a list of available datasets, use the see_datasets command.

II. Visibility

see_datasets()

shows available datasets.

see_logs()

shows the history log.

III. Data Alignment

temporal_alignment(universal_time)

temporally interpolates all data in the data sources to the universal time provided as universal_time (a numpy array).

KDE(dataset_name, t_new)

Performs kernel density estimate on dataset_name using t_new.

IV. Data Fusion

concatenate(**kwargs)

performs concatenation fusion from the data sources, a low-level fusion technique. Returns the results in the active_dataset. Concatenates what it can unless the heights of datasets do not align. Performs action in one of four ways depending on the call.

i. Provide a list of dataset names with keyword "datasets".
ii. Provide a window scheme with keyword arguments "window_length" and "window_overlap". This will concatenate datasets that were created with this window scheme only.
iii. Provide no input arguments but set a window scheme. This will concatenate datasets that were created with this window scheme only.
iv. Provide no input arguments and did not set a window scheme. This will concatenate all available datasets.

V. Decision Making

naive_bayes()

performs naïve bayes classification. Returns Y predictions into the DFE object. Results can be seen using the classification_report command.

linear_regression()

performs linear regression. Returns Y predictions into the DFE object. Results can be seen using the regression_report command.

LASSO(alpha)

performs LASSO regression. Returns Y prediction into the DFE object. Results can be seen using the regression_report command.

my_LDA()

performs LDA classification. Returns Y prediction into the DFE object. Results can be seen using the classification_report command.

random_forest()

performs Random Forest classification. Returns Y prediction into the DFE object. Results can be seen using the classification_report command.

classification_report()

shows a classification report for latest performed classification method. Includes precision, recall, f1 score, support, and confusion matrix.

regression_report()

shows a regression report for latest performed regression method. Includes Pearson Coefficient, RMSE, Relative RMSE (RRMSE), Mean Absolute Error (MAE), relative absolute error (RAE), and Y predicted vs Y test scatter plot.

VI. Dimension Reduction

my_PCA(dataset_name, n_components=5)

performs PCA on dataset indicated by dataset_name. Optionally can set a different number of components to return but defaults to 5.

my_ICA(dataset_name, n_component=5)

performs ICA on dataset indicated by dataset_name. Optionally can set a different number of components to return but defaults to 5.

VII. Feature Extraction

Temporal feature extraction requires that a window scheme be set first, then you can call one of the feature methods. Each will return the feature data as a new Raw Data Source. For example, calling

```
  >>> dfeo.fe_average("Entry_0")
```

will create a new dataset called “Entry_0_average”.

set_window_scheme(**kwargs)

> sets the windowing scheme used in future temporal feature extractions. Optional keyword arguments are window length (window_length, as positive real number) and window overlap (window_overlap, as positive real number).

fe_average(dataset_name)

> performs temporal feature extraction using averaging on dataset specified as dataset_name.

fe_variance(dataset_name)

> performs temporal feature extraction using variance on dataset specified as dataset_name.

fe_skewness(dataset_name)

> performs temporal feature extraction using skewness on dataset specified as dataset_name.

fe_kurtosis(dataset_name)

> performs temporal feature extraction using kutosis on dataset specified as dataset_name.

fe_peak_count(dataset_name)

> performs temporal feature extraction using peak count on dataset specified as dataset_name.

fe_RMS(dataset_name)

> performs temporal feature extraction using root mean square on dataset specified as dataset_name.

classification_windowing(datset_name)

> performs temporal feature extraction using majority classification on dataset specified as dataset_name.

normalize(datset_name, ignore = [], type=’Z’)

> normalizes data in dataset_name. Defaults type to Z-score normalization (‘Z’) but can be set to min-max normalization (‘MinMax’). Will ignore column indices specified in ignore.