tsfresh Documentation

Release

Nov 04, 2016

Contents

This is the documentation of **tsfresh**.

tsfresh is a python package that is used to automatically calculate a huge number of time series characteristics, the so called features. Further the package contains methods to evaluate the explaining power and importance of such characteristics for regression or classification tasks.

CHAPTER 1

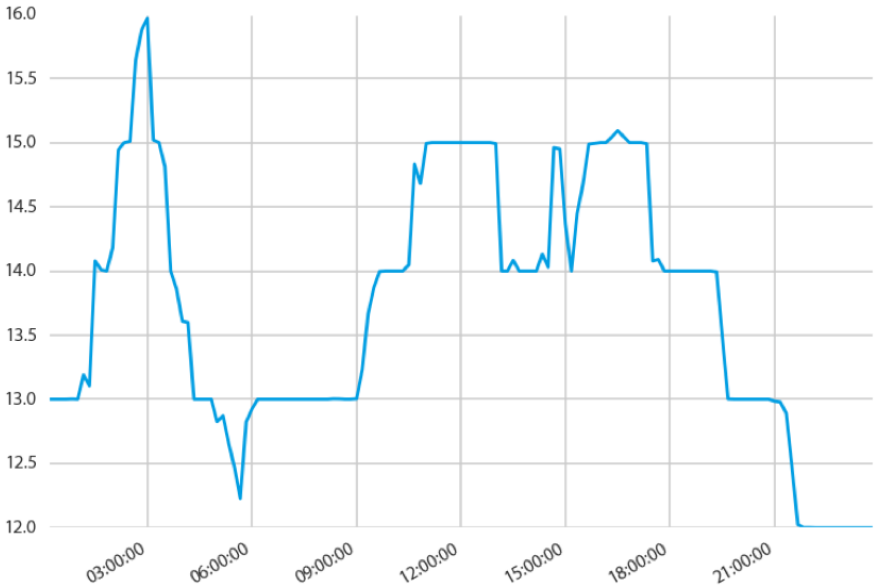
Contents

The following chapters will explain the tsfresh package in detail:

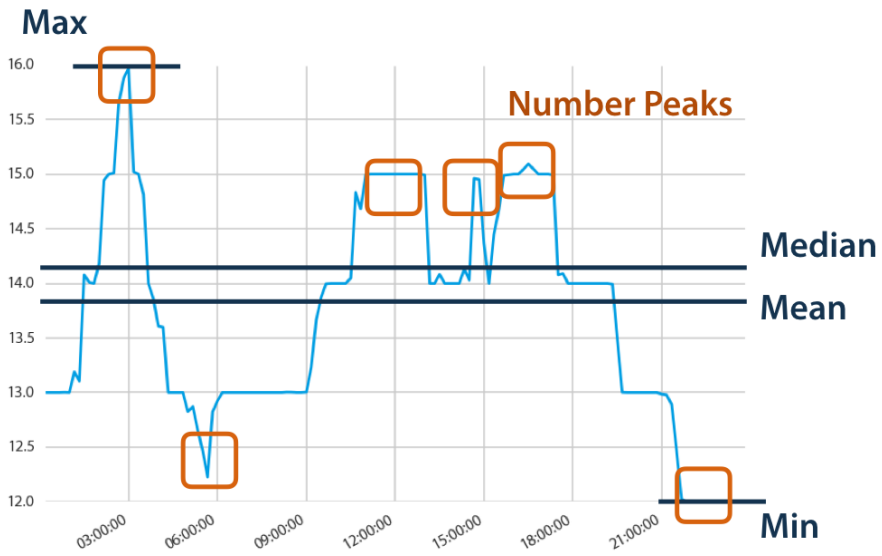
1.1 Introduction

1.1.1 Why do you need such a module?

tsfresh is used to to extract characteristics from time series. Let's assume you recorded the ambient temperature around your computer over one day as the following time series:



Now you want to calculate different characteristics such as the maximal or minimal temperature, the average temperature or the number of temporary temperature peaks:



Without tsfresh, you would have to calculate all those characteristics by hand. With tsfresh this process is automated and all those features can be calculated automatically.

Further tsfresh is compatible with pythons `pandas` and `scikit-learn` APIs, two important packages for Data Science endeavours in python.

1.1.2 What to do with these features?

The extracted features can be used to describe or cluster time series based on the extracted characteristics. Further, they can be used to build models that perform classification/regression tasks on the time series. Often the features give new insights into time series and their dynamics.

The tsfresh package has been used successfully in projects involving

- the prediction of the life span of machines
- the prediction of the quality of steel billets during a continuous casting process

1.1.3 What not to do with tsfresh?

Currently, tsfresh is not suitable

- for usage with streaming data
- for batch processing over a distributed architecture, where different time series are fragmented over different computational units
- to train models on the features (we do not want to reinvent the wheel, check out the python package scikit-learn for example)

However, some of these use cases could be implemented, if you have an application in mind, open an issue at https://github.com/blue-yonder, or feel free to contact us.

1.1.4 What else is out there?

There is a matlab package called hctsa which can be used to automatically extract features from time series. It is also possible to use hctsa from within python by means of the pyopy package.

1.2 Quick Start

1.2.1 Install tsfresh

As the compiled tsfresh package is hosted on pypy you can easily install it with pip

```
pip install tsfresh
```

1.2.2 Dive in

Before boring yourself by reading the docs in detail, you can dive right into tsfresh with the following example:

We are given a data set containing robot failures as discussed in [1]. Each robot records time series from six different sensors. For each sample denoted by a different id we are going to classify if the robot reports a failure or not. From a machine learning point of view, our goal is to classify each group of time series.

To start, we load the data into python

```
from tsfresh.examples import load_robot_execution_failures
timeseries, y = load_robot_execution_failures()
```

and end up with a pandas.DataFrame *timeseries* having the following shape

	id	time	a	b	c	d	e	f
0	1	0	-1	-1	63	-3	-1	0
1	1	1	0	0	62	-3	-1	0
2	1	2	-1	-1	61	-3	0	0
3	1	3	-1	-1	63	-2	-1	0
4	1	4	-1	-1	63	-3	-1	0
...	...	...	...	...	...	...	...	...

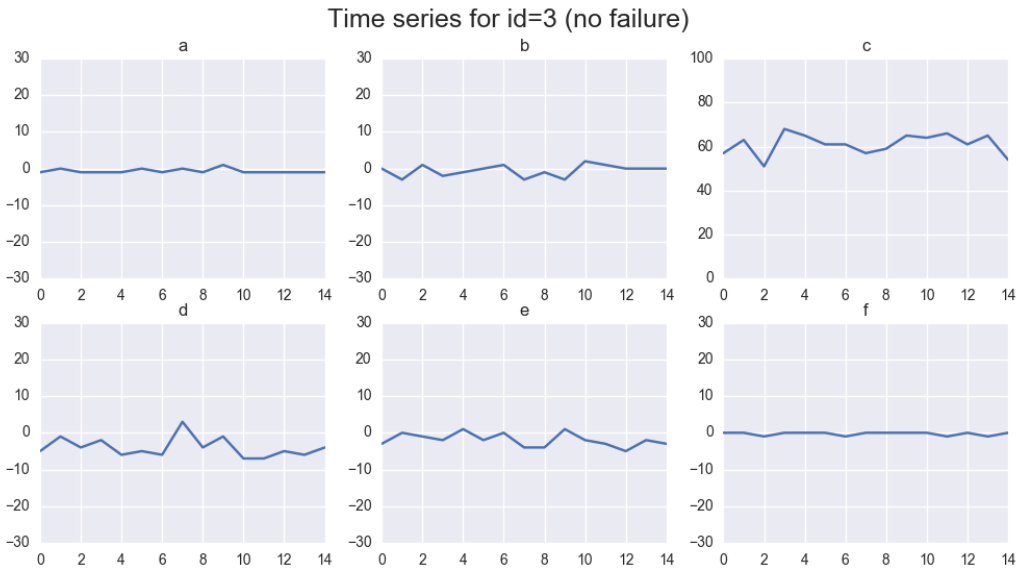
The first column is the DataFrame index and has no meaning here. There are six different time series (a-f) for all different robots that are denoted by the ids column.

On the other hand, `y` contains the information which id belongs to a failure and which not:

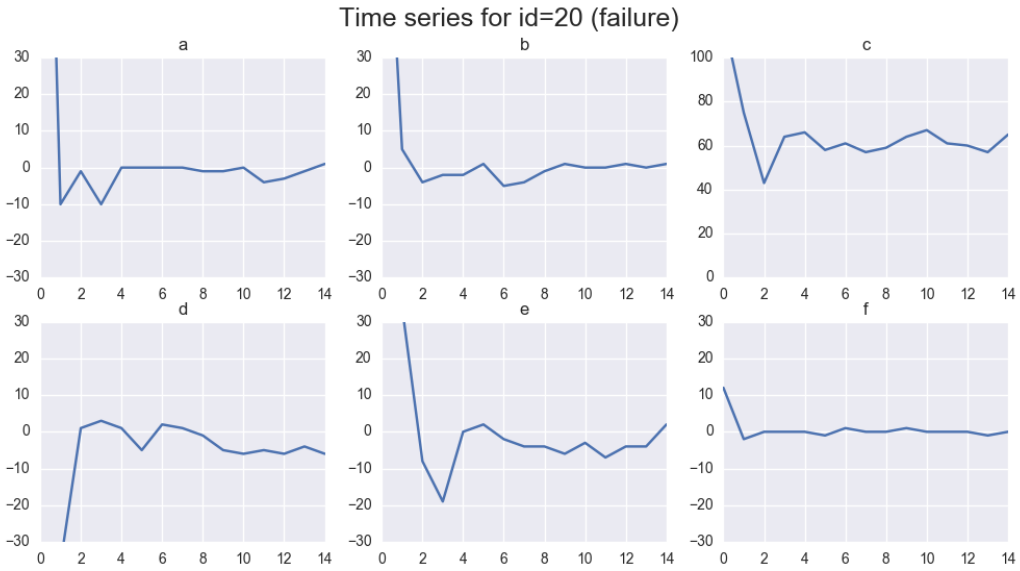
1	0
2	0
3	0
4	0
5	0
...	...

Here, for the samples with ids 1 to 5 no failure was reported.

In the following we illustrate the time series of the sample id 3 reporting no failure:



And for id 20 reporting a failure:



Now tsfresh comes into place. It allows us to automatically extract over 1200 features from those six different time series.

First we extract all features:

```
from tsfresh import extract_features
extracted_features = extract_features(timeseries, column_id="id", column_sort="time")
```

You end up with all extracted features, which are more than 1200 different. We will now remove all `NaN` values and select only the relevant features

```
from tsfresh import select_features
from tsfresh.utilities.dataframe_functions import impute
```

```
impute(extracted_features)
features_filtered = select_features(extracted_features, y)
```

Only around 300 features were classified as relevant enough.

Further, you can even perform the extraction, imputing and filtering at the same time with the `tsfresh.convenience.extract_relevant_features()` function:

```
from tsfresh import extract_relevant_features

features_filtered_direct = extract_relevant_features(timeseries, y, column_id='id',␣
↪column_sort='time')
```

You can now use the features contained in the Data Frame *features_filtered* (which is equal to *features_filtered_direct*) in conjunction with *y* to train your model. Please see the *robot_failure_example.ipynb* Jupyter Notebook in the folder named notebook. In this notebook a RandomForestClassifier is trained on the extracted features.

References

1.3 tsfresh

1.3.1 tsfresh package

Subpackages

tsfresh.convenience package

Submodules

tsfresh.convenience.relevant_extraction module

autodoc: failed to import module u'tsfresh.convenience.relevant_extraction'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

Module contents

autodoc: failed to import module u'tsfresh.convenience'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-

packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.examples package

Submodules

tsfresh.examples.robot_execution_failures module

autodoc: failed to import module u'tsfresh.examples.robot_execution_failures'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

Module contents

autodoc: failed to import module u'tsfresh.examples'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.feature_extraction package

Submodules

tsfresh.feature_extraction.extraction module

autodoc: failed to import module u'tsfresh.feature_extraction.extraction'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.feature_extraction.feature_calculators module

autodoc: failed to import module u'tsfresh.feature_extraction.feature_calculators'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.feature_extraction.settings module

autodoc: failed to import module u'tsfresh.feature_extraction.settings'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

Module contents

autodoc: failed to import module u'tsfresh.feature_extraction'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.feature_selection package

Submodules

tsfresh.feature_selection.feature_selector module

autodoc: failed to import module u'tsfresh.feature_selection.feature_selector'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.feature_selection.selection module

autodoc: failed to import module u'tsfresh.feature_selection.selection'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.feature_selection.settings module

autodoc: failed to import module u'tsfresh.feature_selection.settings'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.feature_selection.significance_tests module

autodoc: failed to import module u'tsfresh.feature_selection.significance_tests'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

Module contents

autodoc: failed to import module u'tsfresh.feature_selection'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.transformers package

Submodules

tsfresh.transformers.feature_augmenter module

autodoc: failed to import module u'tsfresh.transformers.feature_augmenter'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.transformers.feature_selector module

autodoc: failed to import module u'tsfresh.transformers.feature_selector'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.transformers.relevant_feature_augmenter module

autodoc: failed to import module u'tsfresh.transformers.relevant_feature_augmenter'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

Module contents

autodoc: failed to import module u'tsfresh.transformers'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.utilities package

Submodules

tsfresh.utilities.dataframe_functions module

autodoc: failed to import module u'tsfresh.utilities.dataframe_functions'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

tsfresh.utilities.profiling module

autodoc: failed to import module u'tsfresh.utilities.profiling'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

Module contents

autodoc: failed to import module u'tsfresh.utilities'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

Module contents

autodoc: failed to import module u'tsfresh'; the following exception was raised: Traceback (most recent call last): File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/sphinx/ext/autodoc.py", line 519, in import_object __import__(self.modname) File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/__init__.py", line 21, in <module> from tsfresh.convenience.relevant_extraction import extract_relevant_features File "/home/docs/checkouts/readthedocs.org/user_builds/tsfresh/envs/v0.1.2/local/lib/python2.7/site-packages/tsfresh/convenience/relevant_extraction.py", line 6, in <module> import pandas as pd ImportError: No module named pandas

1.4 Data Formats

tsfresh offers three different options to specify the time series data to be used. Irrespective of the input format, tsfresh will always return the calculated features in the same output format.

All three input format options consist of `pandas.DataFrame` objects. There are four important column types that make up those DataFrames:

Mandatory

column_id This column indicates which entities the time series belong to. Features will be extracted individually for each entity. The resulting feature matrix will contain one row per entity.

Optional (but strongly recommended to specify)

column_sort This column contains values which allow to sort the time series (e.g. time stamps). It is not required to have equidistant time steps or the same time scale for the different ids and/or kinds. If you omit this column, the DataFrame is assumed to be sorted in increasing order.

Optional

column_kind This column indicates the names of the different time series types (E.g. different sensors in an industrial application). For each kind of time series the features are calculated individually.

column_value This column contains the actual values of the time series.

Important: None of these columns is allowed to contain any `NaN`, `Inf` or `-Inf` values.

Now there are three slightly different input formats for the time series data:

- A flat DataFrame
- A stacked DataFrame
- A dictionary of flat DataFrames

The difference between a flat and a stacked DataFrame is indicated by specifying or not specifying the parameters *column_value* and *column_kind*.

1.4.1 Input Option 1. Flat DataFrame

If both *column_value* and *column_kind* are set to `None`, the time series data is assumed to be in a flat DataFrame. This means that each different time series is saved as its own column.

Example: Imagine you record the values of time series x and y for different objects A and B for three different times t1, t2 and t3. Now you want to calculate some feature with tsfresh. Your resulting DataFrame have to look like this:

id	time	x	y
A	t1	x(A, t1)	y(A, t1)
A	t2	x(A, t2)	y(A, t2)
A	t3	x(A, t3)	y(A, t3)
B	t1	x(B, t1)	y(B, t1)
B	t2	x(B, t2)	y(B, t2)
B	t3	x(B, t3)	y(B, t3)

and you would pass

```
column_id="id", column_sort="time", column_kind=None, column_value=None
```

to the extraction functions.

1.4.2 Input Option 2. Stacked DataFrame

If both *column_value* and *column_kind* are set, the time series data is assumed to be a stacked DataFrame. This means that there are no different columns for the different type of time series. This representation has several advantages over the flat Data Frame. For example, the time stamps of the different time series do not have to align.

In the above example, you can leave out the time column in the DataFrame (and also in the parameters) and pass a stacked DataFrame.

It does not contain different columns for the different types of time series but only one value column and a kind column:

id	time	kind	value
A	t1	x	x(A, t1)
A	t2	x	x(A, t2)
A	t3	x	x(A, t3)
A	t1	y	y(A, t1)
A	t2	y	y(A, t2)
A	t3	y	y(A, t3)
B	t1	x	x(B, t1)
B	t2	x	x(B, t2)
B	t3	x	x(B, t3)
B	t1	y	y(B, t1)
B	t2	y	y(B, t2)
B	t3	y	y(B, t3)

Then you would set

```
column_id="id", column_sort="time", column_kind="kind", column_value="value"
```

1.4.3 Input Option 3. Dictionary of flat DataFrames

Instead of passing a DataFrame which must be split up by its different kinds, you can also give a dictionary mapping from the kind as string to a DataFrame containing only the time series data of that kind. So essentially you are using a singular DataFrame for each kind of time series.

The data from the example can be split into two DataFrames resulting in the following dictionary

{ “x”:

id	time	value
A	t1	x(A, t1)
A	t2	x(A, t2)
A	t3	x(A, t3)
B	t1	x(B, t1)
B	t2	x(B, t2)
B	t3	x(B, t3)

, “y”:

id	time	value
A	t1	y(A, t1)
A	t2	y(A, t2)
A	t3	y(A, t3)
B	t1	y(B, t1)
B	t2	y(B, t2)
B	t3	y(B, t3)

}

tsfresh would be passed this dictionary and the following arguments

```
column_id="id", column_sort="time", column_kind=None, column_value="value":
```

In this case we do not need to specify the kind column as the kind is the respective dictionary key.

1.4.4 Output Format

The resulting feature matrix for all three input options will be the same. It will always be a `pandas.DataFrame` with the following layout

id	x_feature_1	...	x_feature_N	y_feature_1	...	y_feature_N
A	...	...	...	...	...	...
B	...	...	...	...	...	...

where the x features are calculated using all x values (independently for A and B), y features using all y values and so on.

1.5 Feature naming

tsfresh enforces a strict naming of the created features. This is due to the `tsfresh.feature_extraction.FeatureExtractionSettings.from_columns()` method which needs to deduce the following information from the feature name

- the time series that was used to calculate the feature
- the feature calculator method that was used to derive the feature
- all parameters that have been used to calculate the feature (optional)

Hence, to enable the `tsfresh.feature_extraction.FeatureExtractionSettings.from_columns()` to deduce all the necessary conditions, the features will be named in the following format

> {time_series_name}__{feature_name}__{parameter name 1}_{parameter value 1}__[..]__{parameter name k}_{parameter value k}

(Here we assumed that {feature_name} has k parameters).

1.5.1 Examples

So for example the following feature name

> temperature_1__quantile__q_0.6

is the value of the feature `tsfresh.feature_extraction.feature_calculators.quantile()` for the time series `` `temperature_1` `` and a parameter value of `q=0.6`. On the other hand, the feature named

> Pressure 5__cwt_coefficients__widths_(2, 5, 10, 20)__coeff_14__w_5

denotes the value of the feature `tsfresh.feature_extraction.feature_calculators.cwt_coefficients()` for the time series `` `Pressure 5` `` under parameter values of `widths=(2,5,10,20)`, `coeff=14` and `w=5`.

1.6 scikit-learn Transformers

tsfresh includes three scikit-learn compatible transformers. You can easily add them to your existing data science pipeline. If you are not familiar with scikit-learn's pipeline we recommend you take a look at the official documentation [1].

The purpose of such a pipeline is to assemble several preprocessing steps that can be cross-validated together while setting different parameters. Our tsfresh transformer allows you to extract and filter the time series features during such a preprocessing sequence.

The first two estimator contained in tsfresh are the `FeatureAugmenter`, which extracts the features, and the `FeatureSelector`, which only performs the feature selection algorithm. Is preferable to combine extracting and filtering of the features in a single step to avoid unnecessary feature calculations. Hence, we have the `RelevantFeatureAugmenter`, which combines both the extraction and filtering of the features in a single step.

1.6.1 Example

In the following example you see how we combine tsfresh's `RelevantFeatureAugmenter` and a `RandomForestClassifier` into a single pipeline. This pipeline can then fit both our transformer and the classifier in one step.

```
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from tsfresh.examples import load_robot_execution_failures
from tsfresh.transformers import RelevantFeatureAugmenter

pipeline = Pipeline([('augmenter', RelevantFeatureAugmenter(column_id='id', column_
↪sort='time')),
            ('classifier', RandomForestClassifier())])

df_ts, y = load_robot_execution_failures()
X = pd.DataFrame(index=y.index)

pipeline.set_params(augmenter__timeseries_container=df_ts)
pipeline.fit(X, y)
```

The parameters of the augment transformer correspond to the parameters of the top-level convenience function `extract_relevant_features()`. In the example, we only set the names of two columns `column_id='id'`, `column_sort='time'` (see *Data Formats* for an explanation of those parameters).

Because we can not pass the time series container directly as a parameter to the augmenter step when calling fit or transform on a `sklearn.pipeline.Pipeline` we have to set it manually by calling `pipeline.set_params(augmenter__timeseries_container=df_ts)`. In general, you can change the time series container from which the features are extracted by calling either the pipeline's `set_params()` method or the transformers `set_timeseries_container()` method.

For further examples, see the Jupyter Notebook pipeline_example.ipynb in the notebooks folder of the tsfresh package.

1.6.2 References

[1] http://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html

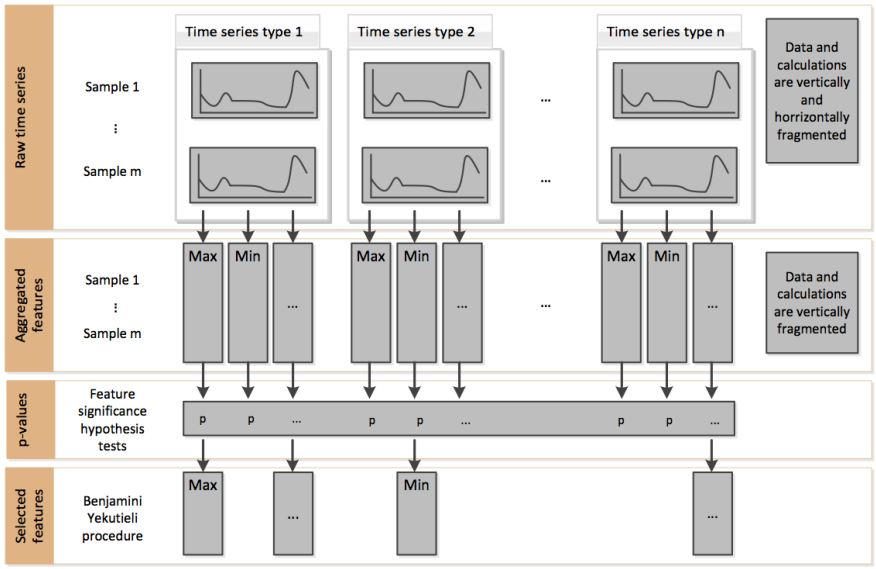
1.7 Feature filtering

The all-relevant problem of feature selection is the identification of all strongly and weakly relevant attributes. This problem is especially hard to solve for time series classification and regression in industrial applications such as predictive maintenance or production line optimization, for which each label or regression target is associated with several time series and meta-information simultaneously.

To limit the number of irrelevant features, tsfresh deploys the fresh algorithm (fresh stands for *FeatuRe Extraction based on Scalable Hypothesis tests*) [1].

The algorithm is called by `tsfresh.feature_selection.feature_selector.check_fs_sig_bh()`. It is a efficient, scalable feature extraction algorithm, which filters the available features in an early stage of the machine learning pipeline with respect to their significance for the classification or regression task, while controlling the expected percentage of selected but irrelevant features.

The filtering process consists of three phases which are sketched in the following figure:



1.7.1 Phase 1 - Feature extraction

Firstly, the algorithm characterizes time series with comprehensive and well-established feature mappings and considers additional features describing meta-information. The feature calculators used to derive the features are contained in `tsfresh.feature_extraction.feature_calculators`.

In the figure from above, this corresponds to the change from raw time series to aggregated features.

1.7.2 Phase 2 - Feature significance testing

In a second step, each feature vector is individually and independently evaluated with respect to its significance for predicting the target under investigation. Those tests are contained in the submodule `tsfresh.feature_selection.significance_tests`. The result of these tests is a vector of p-values, quantifying the significance of each feature for predicting the label/target.

In the figure from above, this corresponds to the change from aggregated features to p-values.

[1] Christ, M., Kempa-Liehr, A.W. and Feindt, M. (2016). Distributed and parallel time series feature extraction for industrial big data applications. ArXiv e-prints: 1610.07717 URL: http://adsabs.harvard.edu/abs/2016arXiv161007717C

1.7.3 Phase 3 - Multiple test procedure

The vector of p-values is evaluated on basis of the Benjamini-Yekutieli procedure [2] in order to decide which features to keep. This multiple testing procedure is contained in the submodule `tsfresh.feature_selection.feature_selector`.

In the figure from above, this corresponds to the change from p-values to selected features.

1.7.4 References

1.8 How to add a custom feature

It may be beneficial to add a custom feature to those that are calculated by tsfresh. To do so, one has to adapt certain steps:

1.8.1 Step 1. Decide which type of feature you want to implement

In tsfresh we differentiate between three types of feature calculation methods

1. aggregate features without parameter
2. aggregate features with parameter
3. apply features with parameters

So if you want to add a singular feature with out any parameters, stick with *1.*, the aggregate feature without parameters.

Then, if your features can be calculated independently for each parameter, stick with type *2.*, the aggregate features with parameters.

If both cases from above do not apply, so it is beneficial to calculate the features for the different parameter settings at the same time (to e.g. perform auxiliary calculations only once for all features), stick with type *3.*, the apply features with parameters.

1.8.2 Step 2. Write the feature calculator

Depending on which type of feature you are implementing, you can use the following feature calculator skeletons:

1. aggregate features without parameter

```
@set_property("fctype", "aggregate")
def your_feature_calculator(x):
    """
    The description of your feature

    :param x: the time series to calculate the feature of
    :type x: pandas.Series
    :return: the value of this feature
    :return type: bool or float
    """
```

[2] Benjamini, Y. and Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. Annals of statistics, 1165–1188

```
    # Calculation of feature as float, int or bool
    f = f(x)
    return f
```

2. aggregate features with parameter

```
@set_property("fctype", "aggregate_with_parameters")
def your_feature_calculator(x, p1, p2, ...):
    """
    Description of your feature

    :param x: the time series to calculate the feature of
    :type x: pandas.Series
    :param p1: description of your parameter p1
    :type p1: type of your parameter p1
    :param p2: description of your parameter p2
    :type p2: type of your parameter p2
    ...
    :return: the value of this feature
    :return type: bool or float
    """
    # Calculation of feature as float, int or bool
    f = f(x)
    return f
```

3. apply features with parameters

```
@set_property("fctype", "apply")
def your_feature_calculator(x, c, param):
    """
    Description of your feature

    :param x: the time series to calculate the feature of
    :type x: pandas.Series
    :param c: the time series name
    :type c: str
    :param param: contains dictionaries {"p1": x, "p2": y, ...} with p1 float, p2 int␣
↪...
    :type param: list
    :return: the different feature values
    :return type: pandas.Series
    """
    # Calculation of feature as pandas.Series s, the index is the name of the feature
    s = f(x)
    return s
```

After implementing the feature calculator, please add it to the `tsfresh.feature_extraction.extraction` submodule. tsfresh will only find feature calculators that are in this submodule.

1.8.3 Step 3. Add custom settings for your feature

Finally, you have to add custom settings if your feature is a apply or aggregate feature with parameters. To do so, just append your parameters to the `name_to_param` dictionary inside the `tsfresh.feature_extraction.settings.set_default_parameters()` method:

```
name_to_param.update({
    # here are the existing settings
    ...
    # Now the settings of your feature calculator
    "your_feature_calculator" = [{"p1": x, "p2": y, ...} for x,y in ...],
})
```

That is it, tsfresh will calculate your feature the next time you run it.

1.9 Authors

1.9.1 Development Lead

- Maximilian Christ (maximilianchrist.com)

1.9.2 Contributions

- Nils Braun (nilslennartbraun@gmail.com)
- Julius Neuffer (julius.neuffer@blue-yonder.com)

1.10 License

```
MIT LICENCE

Copyright (c) 2016 Maximilian Christ, Blue Yonder GmbH

Permission is hereby granted, free of charge, to any person obtaining a copy of this␣
 ↪software and associated
documentation files (the "Software"), to deal in the Software without restriction,␣
 ↪including without limitation the
rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell␣
 ↪copies of the Software, and to permit
persons to whom the Software is furnished to do so, subject to the following␣
 ↪conditions:

The above copyright notice and this permission notice shall be included in all copies␣
 ↪or substantial portions of the
Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED,␣
 ↪INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.␣
 ↪IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN␣
 ↪ACTION OF CONTRACT, TORT OR
OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR␣
 ↪OTHER DEALINGS IN THE SOFTWARE.
```

1.11 Changelog

tsfresh uses Semantic Versioning

1.11.1 Version 0.1.2

- added support for python 3.5.2
- fixed bug with the naming of the features that made the naming of features non-deterministic

1.11.2 Version 0.1.1

- mainly fixes for the read-the-docs documentation, the pypi readme and so on

1.11.3 Version 0.1.0

- Initial version :)

1.12 How to contribute

We want tsfresh to become the biggest archive of feature extraction methods in python. To achieve this goal, we need your help!

So if you want to add one or two interesting feature calculators, implement a new feature selection process or just fix 1-2 typos, your help is appreciated.

If you want to help, just create a pull request on our github page. If you are not familiar with the versioning tool git you can contact us by email.

We are looking forward to hear from you! =)

CHAPTER 2

Indices and tables

- genindex
- modindex
- search