pyromorphite

Release 0.0.3

Mar 11, 2020

Contents

Pyromorphite is a simple and efficient tool for process mining.

CHAPTER 1

Features

- Import XES or CSV files from local storage or via http
- Multiset (dict) and panda's DataFrame as main log storage abstractions
- Inductive Miner
- Log Skeleton

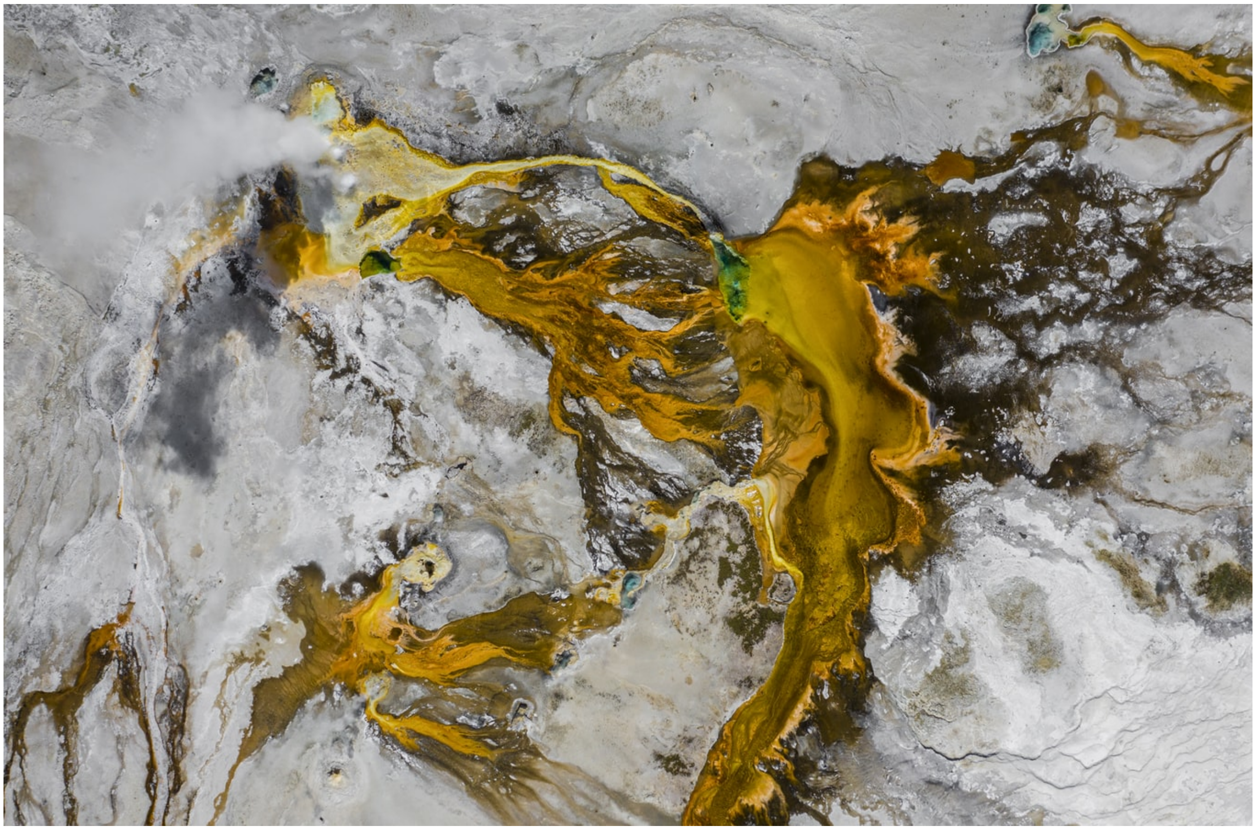
Requests officially supports Python 3.5–3.8.

CHAPTER 2

User Guide

Here you'll get you to setup up pyromorphite and then we'll go over the first few steps you can take with the library.

2.1 Installation



This part of the documentation covers the installation of Pyromorphite. The first step to using any software package is getting it properly installed.

2.1.1 Get the Source Code

Requests is actively developed on GitHub, where the code is always available.

You can either clone the public repository:

```
$ git clone git://github.com/xcavation/pyromorphite.git
```

Once you have a copy of the source, you can embed it in your own Python package, or install it into your site-packages easily:

```
$ cd pyromorphite
$ pip install .
```

2.2 Quickstart

This is a quick introduction to Pyromorphite. Before proceeding, make sure that Pyromorphite is *installed*.

Now let's make sure pyromorphite is imported:

```
>>> import pyromorphite as pm
```

2.2.1 Read a Log

Reading in event log files in Pyromorphite is super easy. It supports XES natively and CSV as well as Excel files via pandas.

XES Files

Having previously imported pyromorphite, we can look for a xes file to import. We'll pick this only repository as it's full with further examples you can try out

```
>>> URL = "https://data.4tu.nl/repository/uuid:c1e9137e-2877-410d-a76a-21ce7f97a239/
↪DATA1"
>>> log = pm.read_xes(URL)
```

CSV and Excel with pandas

Begin by import the pandas module:

```
>>> import pandas as pd
```

We'll try now to get an XES file from a web repository, like this dataset:

```
>>> URL = "https://data.4tu.nl/repository/uuid:d5ccb355-ca67-480f-8739-289b9b593aaf/
↪DATA"
>>> log = pd.read_csv(URL)
```

2.2.2 Construct a Bag

Having parsed a log into a pandas DataFrame we can extract all unique traces together with their frequency in the log as a multiset or bag:

```
>>> bag = pm.as_bag(log)
```

We should also consider that not everyone might use the same column naming in their documents:

```
>>> bag = pm.as_bag(log, case='CI Name (aff)', time='Actual Start', activity='Change␣
↪Type')
```

We can therefore specify the column names to be used as:

- case identifiers
- event label
- and timestamp

2.2.3 Mine Your First Model

With our bag of traces in hand we can go onto mining our first model. We'll go for a process tree. This in not so relevant as most models can be converted between one another. What is relevant, though, is the miner we use. We'll give the Inductive Miner a try:

```
>>> ptree = pm.InductiveMiner(bag)
```

Python Module Index

Index